SCALER

PROJECT

Business Case – Target SQL

Detail Analysis

Submitted by
Name: Gyanpriya Misra
Batch: DSML July22 Beginner 1

CONTENT

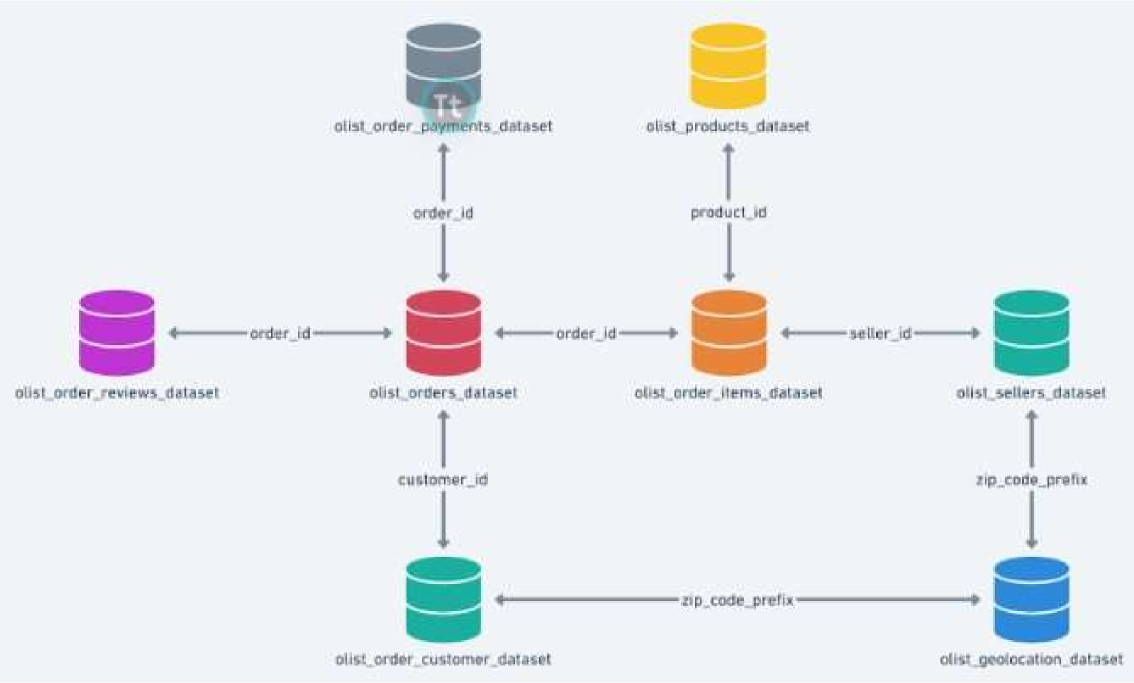
Introduction

Target is one of the world's most recognized brands and one of America's leading retailers. Target makes itself a preferred shopping destination by offering outstanding value, inspiration, innovation and an exceptional guest experience that no other retailer can deliver. This business case has information of 100k orders from 2016 to 2018 made at Target in Brazil. Its features allows viewing an order from multiple dimensions: from order status, price, payment and freight performance to customer location, product attributes and finally reviews written by customers.

Dataset

The dataset is available in 8 tables:
1. customers
2. geolocation
3. order_items
4. payments
5. reviews
6. orders
7. products
8. sellers

Schema



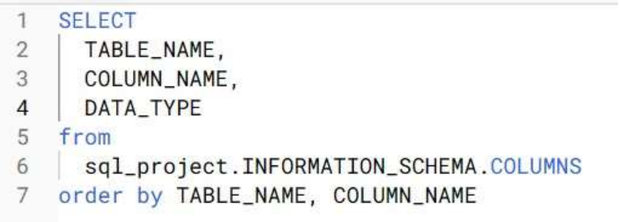
1. IMPORT THE DATASET AND DO USUAL EXPLORATORY ANALYSIS STEPS LIKE CHECKING THE STRUCTURE & CHARACTERISTICS OF THE DATASET.

1.1. Data type of columns in a table

Query:

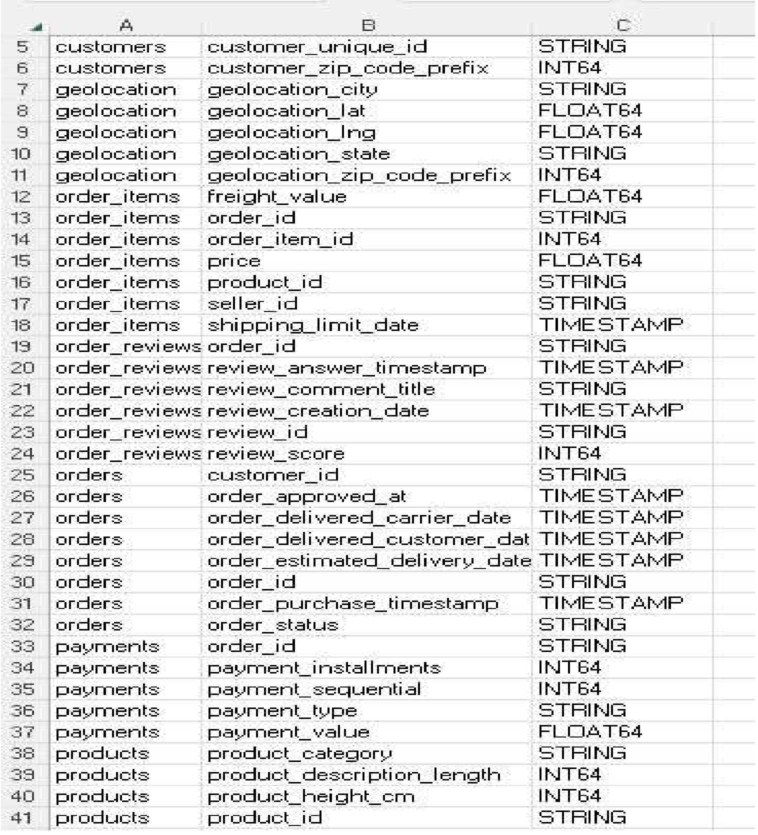
```
SELECT
  TABLE_NAME,
  COLUMN_NAME,
  DATA_TYPE
from
  sql_project.INFORMATION_SCHEMA.COLUMNS
order by TABLE_NAME, COLUMN_NAME
```

Query Screenshot:



```
SELECT
  TABLE_NAME,
  COLUMN_NAME,
  DATA_TYPE
from
  sql_project.INFORMATION_SCHEMA.COLUMNS
order by TABLE_NAME, COLUMN_NAME
```

Result:



	A	B	C
5	customers	customer_unique_id	STRING
6	customers	customer_zip_code_prefix	INT64
7	geolocation	geolocation_city	STRING
8	geolocation	geolocation_lat	FLOAT64
9	geolocation	geolocation_lng	FLOAT64
10	geolocation	geolocation_state	STRING
11	geolocation	geolocation_zip_code_prefix	INT64
12	order_items	freight_value	FLOAT64
13	order_items	order_id	STRING
14	order_items	order_item_id	INT64
15	order_items	price	FLOAT64
16	order_items	product_id	STRING
17	order_items	seller_id	STRING
18	order_items	shipping_limit_date	TIMESTAMP
19	order_reviews	order_id	STRING
20	order_reviews	review_answer_timestamp	TIMESTAMP
21	order_reviews	review_comment_title	STRING
22	order_reviews	review_creation_date	TIMESTAMP
23	order_reviews	review_id	STRING
24	order_reviews	review_score	INT64
25	orders	customer_id	STRING
26	orders	order_approved_at	TIMESTAMP
27	orders	order_delivered_carrier_date	TIMESTAMP
28	orders	order_delivered_customer_dat	TIMESTAMP
29	orders	order_estimated_delivery_date	TIMESTAMP
30	orders	order_id	STRING
31	orders	order_purchase_timestamp	TIMESTAMP
32	orders	order_status	STRING
33	payments	order_id	STRING
34	payments	payment_installments	INT64
35	payments	payment_sequential	INT64
36	payments	payment_type	STRING
37	payments	payment_value	FLOAT64
38	products	product_category	STRING
39	products	product_description_length	INT64
40	products	product_height_cm	INT64
41	products	product_id	STRING

Insight:

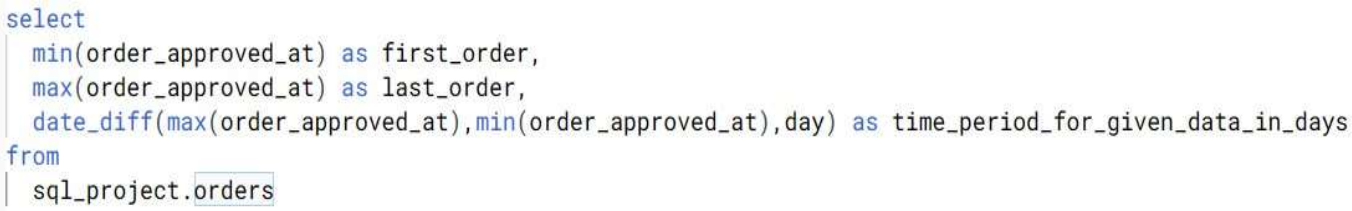
The datatypes used in the tables of dataset are STRING, FLOAT64, TIMESTAMP, INT64 .

1.2. Time period for which the data is given

Query:

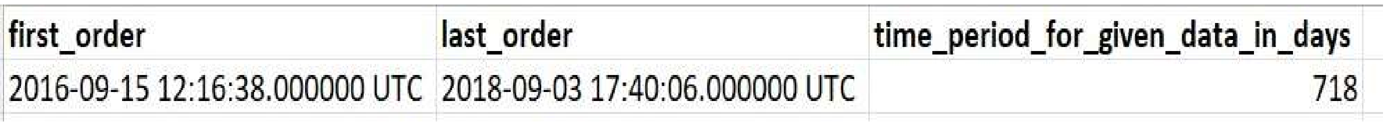
```
select
  min(order_approved_at) as first_order,
  max(order_approved_at) as last_order,
  date_diff(max(order_approved_at),min(order_approved_at),day) as time_p
eriod_for_given_data_in_days
from
  sql_project.orders
```

Query Screenshot:



```
select
  min(order_approved_at) as first_order,
  max(order_approved_at) as last_order,
  date_diff(max(order_approved_at),min(order_approved_at),day) as time_period_for_given_data_in_days
from
  sql_project.orders
```

Result:



first_order	last_order	time_period_for_given_data_in_days
2016-09-15 12:16:38.000000 UTC	2018-09-03 17:40:06.000000 UTC	718

Insight:

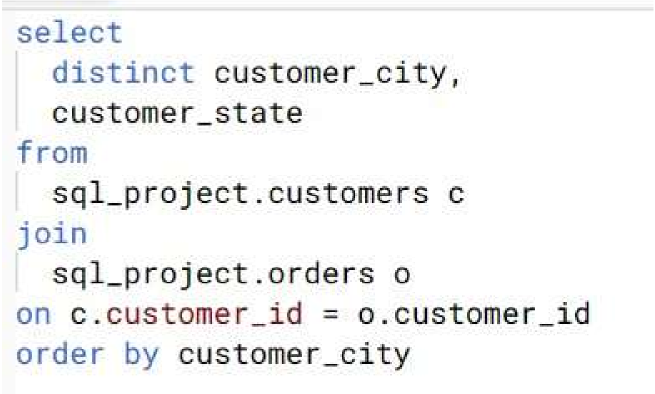
Data of 718 days has been provided for the time period 2016 to 2018.

1.3. Cities and States of customers ordered during the given period

Query:

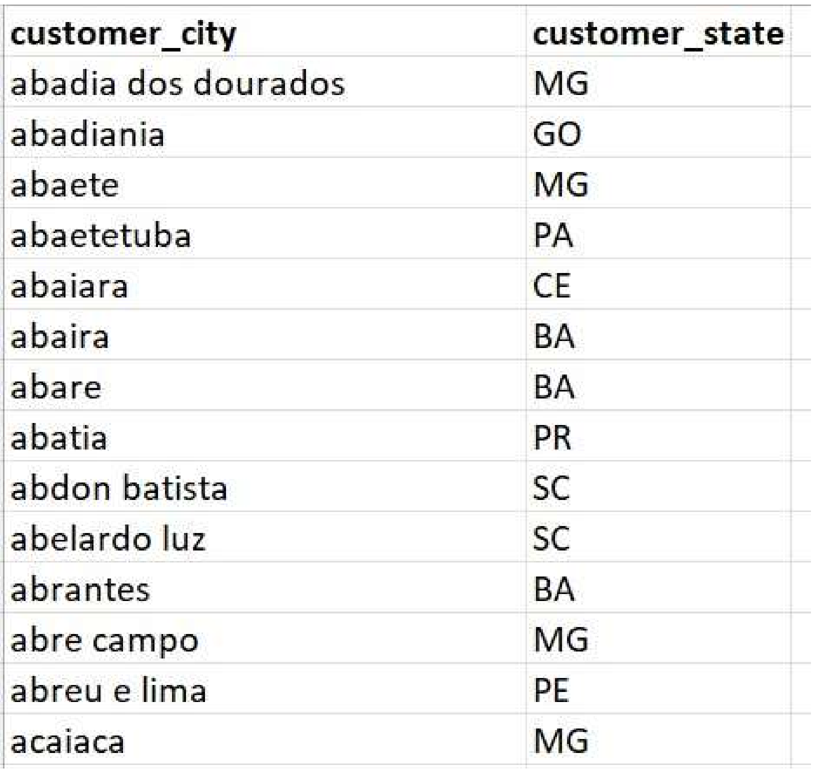
```
select
  distinct customer_city,
  customer_state
from
  sql_project.customers c
join
  sql_project.orders o
on c.customer_id = o.customer_id
order by customer_city
```

Query Screenshot:



```
select
  distinct customer_city,
  customer_state
from
  sql_project.customers c
join
  sql_project.orders o
on c.customer_id = o.customer_id
order by customer_city
```

Result:



customer_city	customer_state
abadia dos dourados	MG
abadiania	GO
abaete	MG
abaetetuba	PA
abaiara	CE
abaira	BA
abare	BA
abatia	PR
abdon batista	SC
abelardo luz	SC
abrantes	BA
abre campo	MG
abreu e lima	PE
acaiaca	MG

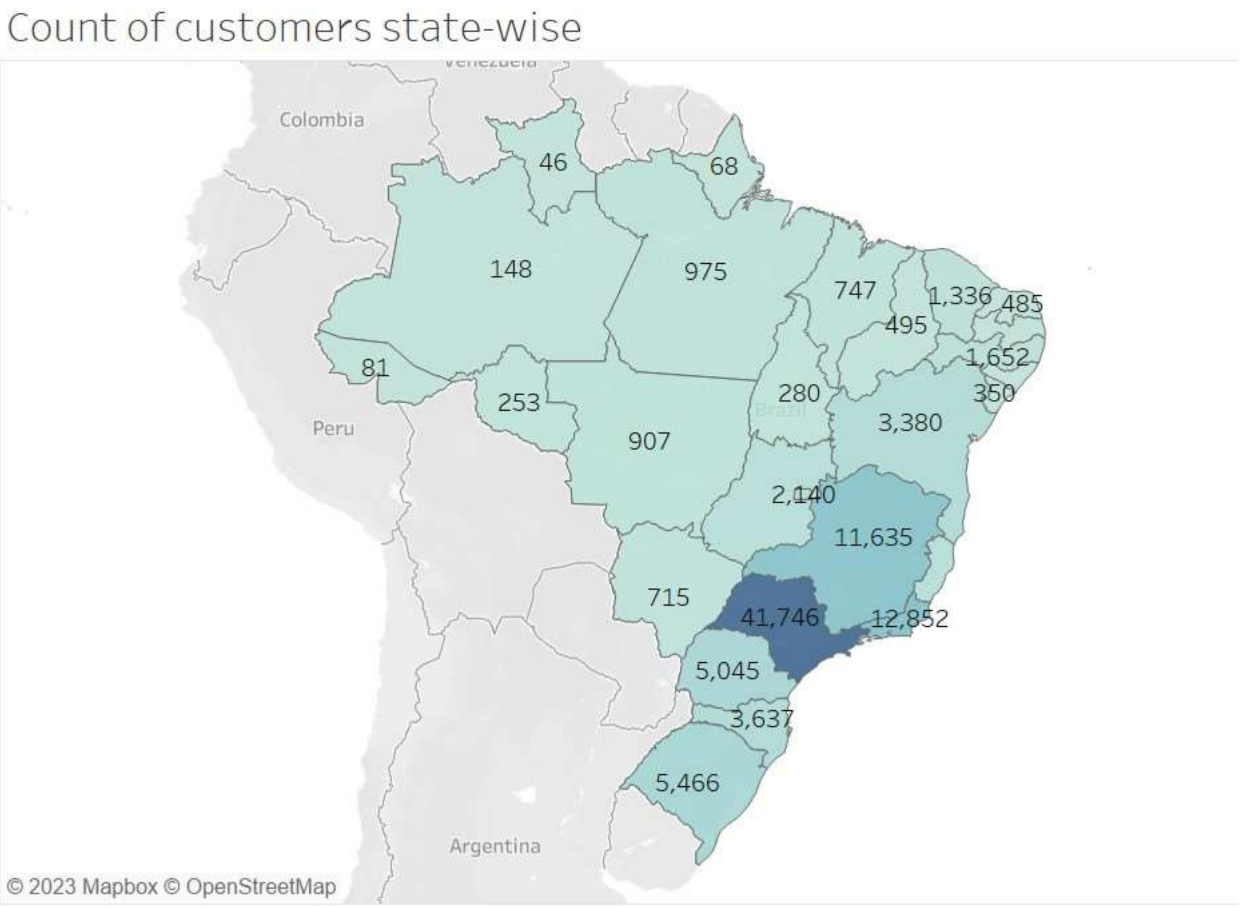
Insights:

Customers belongs to 4310 unique cities and 27 unique states.

Observe the below map showing the distribution of customers over 27 states of Brazil.

It has been observed that Sao Paulo has the highest number of customers 41746 followed by Rio De Janeiro with 12852 and Minas Gerais with 11635.

Roraima in Brazil has lowest customers with count of 46 only.



Count of customers state-wise

© 2023 Mapbox © OpenStreetMap

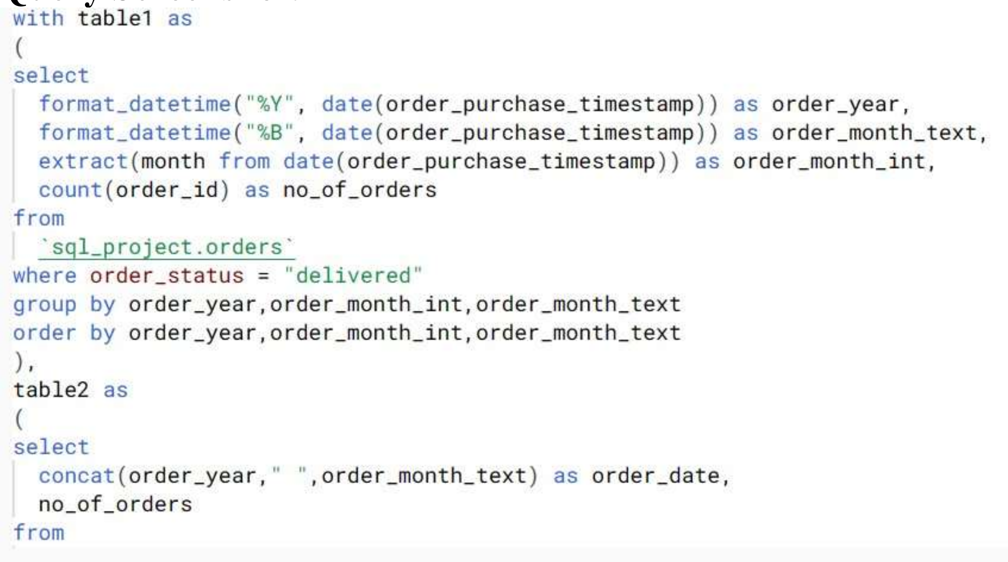
2. IN-DEPTH EXPLORATION:

2.1. Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?

Query:

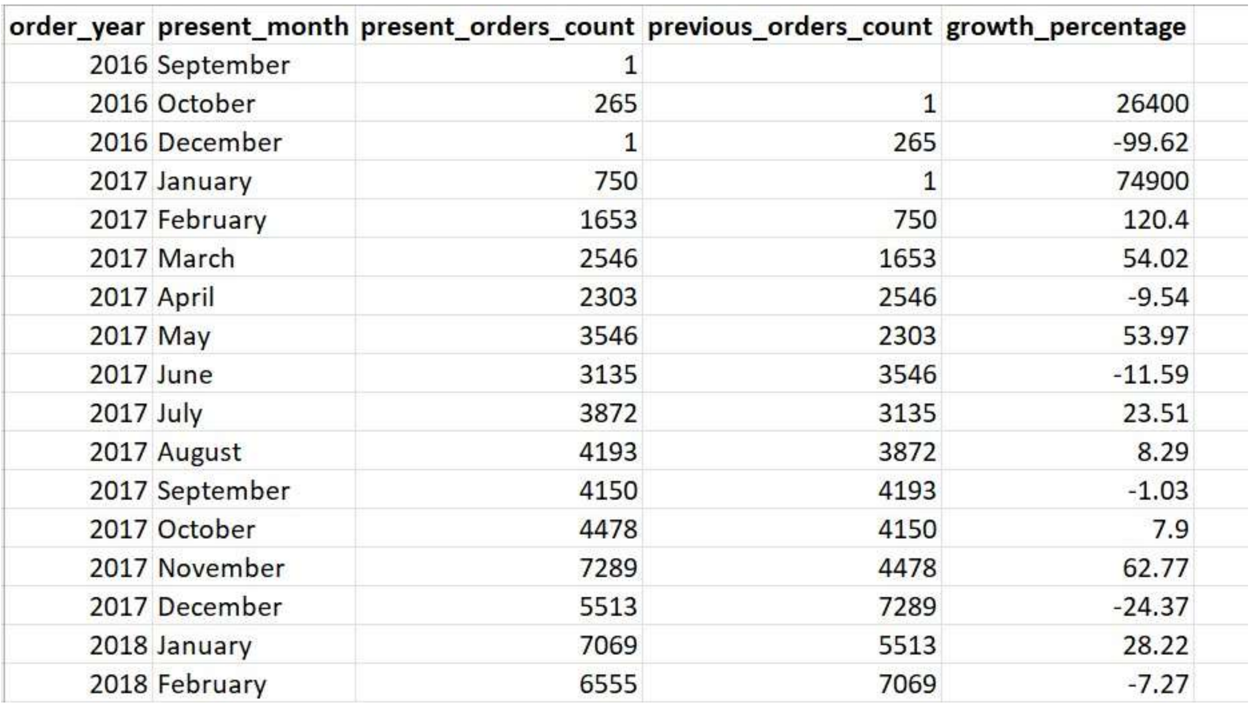
```
with table1 as
(
select
  format_datetime("%Y", date(order_purchase_timestamp)) as order_year,
  format_datetime("%B", date(order_purchase_timestamp)) as order_month_text,
  extract(month from date(order_purchase_timestamp)) as order_month_int,
  count(order_id) as no_of_orders
from
  `sql_project.orders`
where order_status = "delivered"
group by order_year,order_month_int,order_month_text
order by order_year,order_month_int,order_month_text
),
table2 as
(
select
  concat(order_year," ",order_month_text) as order_date,
  no_of_orders
from
 table1
),
table3 as
(
select
  order_year,
  order_month_int,
  order_month_text as present_month,
  no_of_orders as present_orders_count,
  lag(order_month_text) over(order by order_year,order_month_int) as previous_
month,
  lag(no_of_orders) over(order by order_year,order_month_int) as previous_orde
rs_count
from
  table1
order by order_year,order_month_int
)
select
  order_year,
  present_month,
  present_orders_count,
  previous_orders_count,
  round((present_orders_count-
previous_orders_count)/previous_orders_count*100,2) as growth_percentage
from
  table3
```

Query Screenshot:



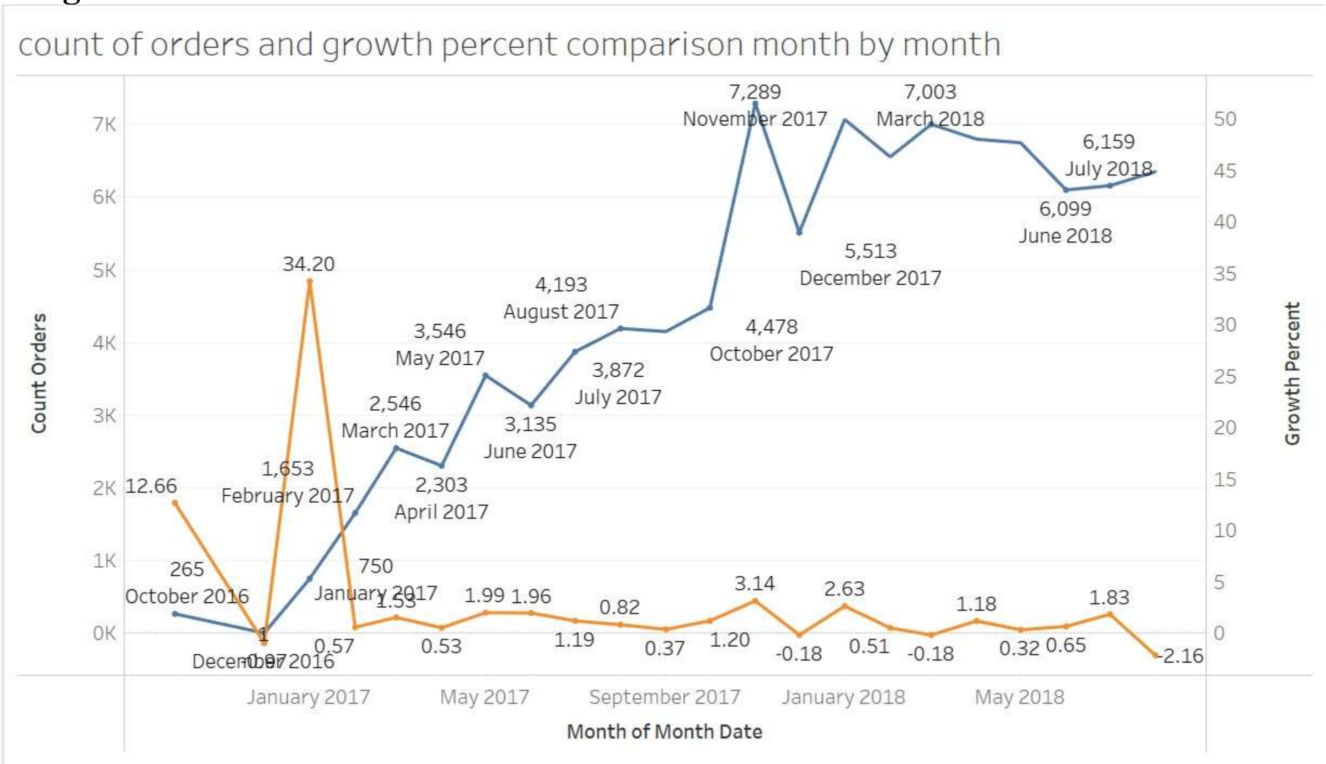
```
with table1 as
(
select
  format_datetime("%Y", date(order_purchase_timestamp)) as order_year,
  format_datetime("%B", date(order_purchase_timestamp)) as order_month_text,
  extract(month from date(order_purchase_timestamp)) as order_month_int,
  count(order_id) as no_of_orders
from
  `sql_project.orders`
where order_status = "delivered"
group by order_year,order_month_int,order_month_text
order by order_year,order_month_int,order_month_text
),
table2 as
(
select
  concat(order_year," ",order_month_text) as order_date,
  no_of_orders
from
```

Result:



order_year	present_month	present_orders_count	previous_orders_count	growth_percentage
2016	September	1		
2016	October	265	1	26400
2016	December	1	265	-99.62
2017	January	750	1	74900
2017	February	1653	750	120.4
2017	March	2546	1653	54.02
2017	April	2303	2546	-9.54
2017	May	3546	2303	53.97
2017	June	3135	3546	-11.59
2017	July	3872	3135	23.51
2017	August	4193	3872	8.29
2017	September	4150	4193	-1.03
2017	October	4478	4150	7.9
2017	November	7289	4478	62.77
2017	December	5513	7289	-24.37
2018	January	7069	5513	28.22
2018	February	6555	7069	-7.27

Insight:



The above graph shows the count of orders month over month from year 2016 to year 2018. It can be observe that the count of orders has been increased in November 2017 with peak count value of 7289. It can be concluded that there is a growing trend in e-commerce.

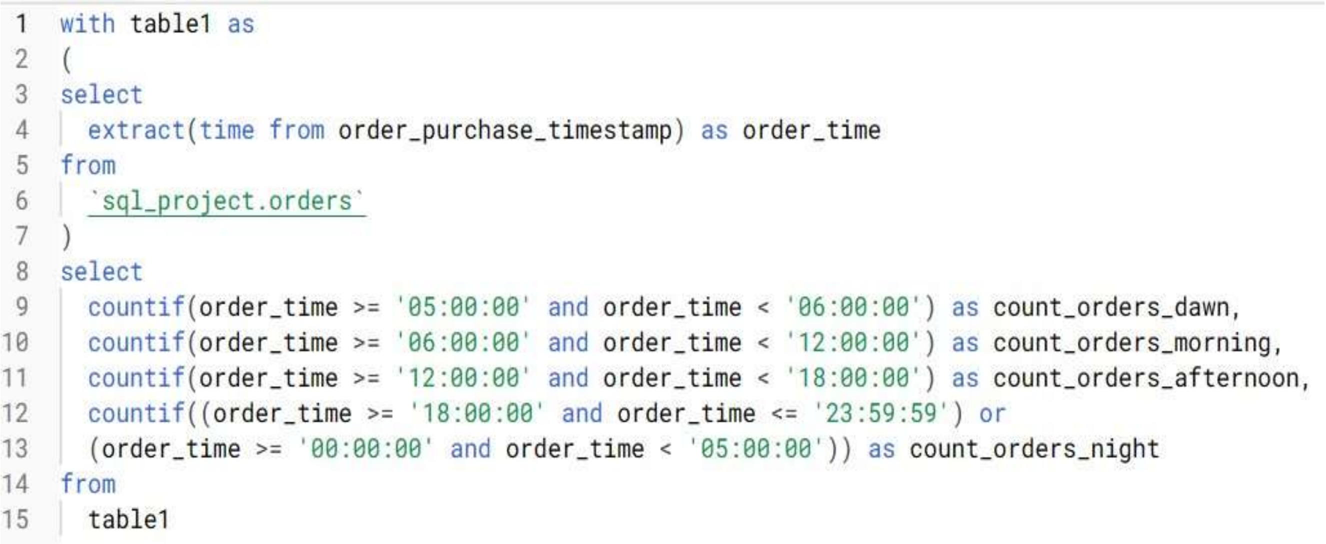
From the complete scenario, it can be analysed that the growth percent of count of orders are not increasing abruptly. It is saturated with positive and negative growth percent over months. However, it has maintained a place in market.

2.2. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

Query:

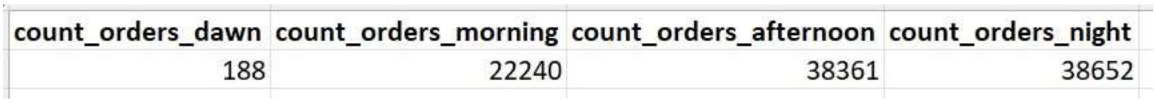
```
with table1 as
(
select
  extract(time from order_purchase_timestamp) as order_time
from
  `sql_project.orders`
)
select
  countif(order_time >= '05:00:00' and order_time < '06:00:00') as count_order
s_dawn,
  countif(order_time >= '06:00:00' and order_time < '12:00:00') as count_order
s_morning,
  countif(order_time >= '12:00:00' and order_time < '18:00:00') as count_order
s_afternoon,
  countif((order_time >= '18:00:00' and order_time <= '23:59:59') or
  (order_time >= '00:00:00' and order_time < '05:00:00')) as count_orders_nigh
t
from
  table1
```

Query Screenshot:



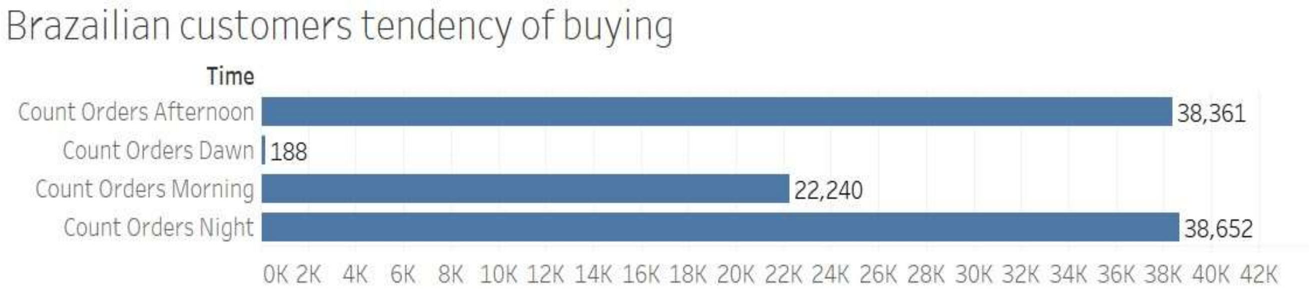
```
with table1 as
(
select
  extract(time from order_purchase_timestamp) as order_time
from
  `sql_project.orders`
)
select
  countif(order_time >= '05:00:00' and order_time < '06:00:00') as count_orders_dawn,
  countif(order_time >= '06:00:00' and order_time < '12:00:00') as count_orders_morning,
  countif(order_time >= '12:00:00' and order_time < '18:00:00') as count_orders_afternoon,
  countif((order_time >= '18:00:00' and order_time <= '23:59:59') or
  (order_time >= '00:00:00' and order_time < '05:00:00')) as count_orders_night
from
  table1
```

Result:



count_orders_dawn	count_orders_morning	count_orders_afternoon	count_orders_night
188	22240	38361	38652

Insights:



Brazilian customers like to shop throughout a day. However, they tend to buy more in night compare with afternoon and morning. Very less (almost negligible) Brazilian customers order at dawn.

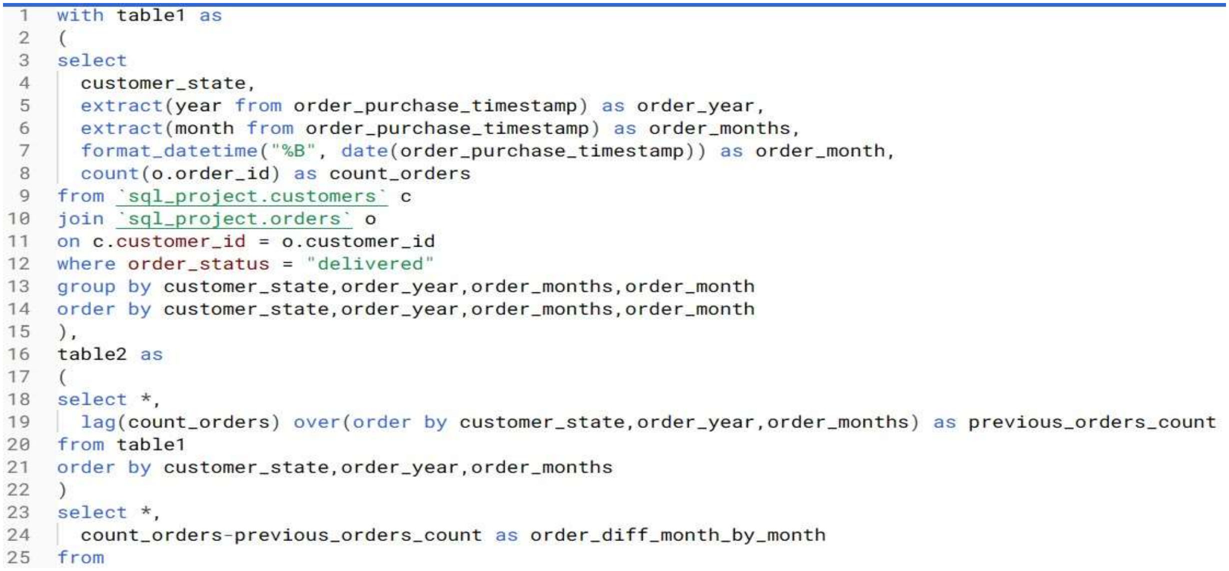
3. EVOLUTION OF E-COMMERCE ORDERS IN THE BRAZIL REGION:

3.1. Get month on month orders by states

Query:

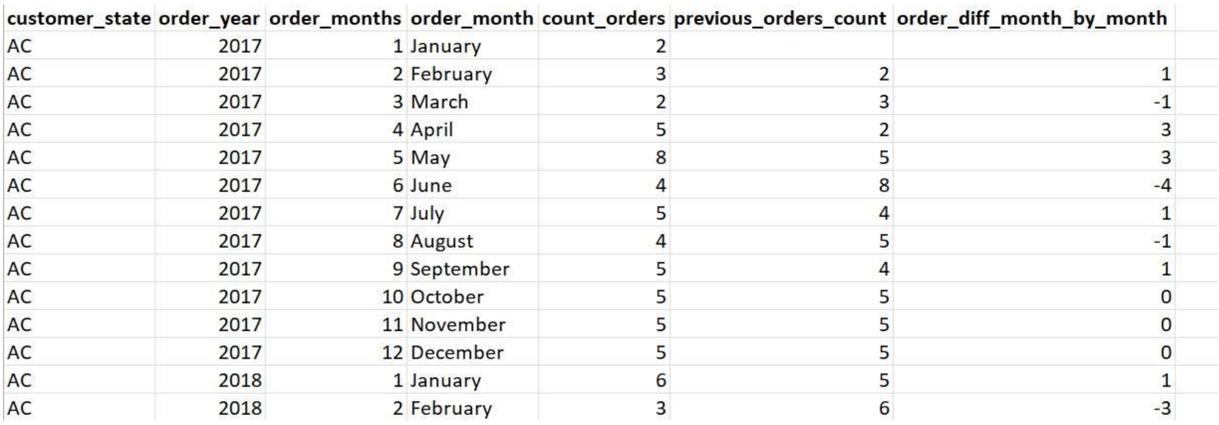
```
with table1 as
(
select
  customer_state,
  extract(year from order_purchase_timestamp) as order_year,
  extract(month from order_purchase_timestamp) as order_months,
  format_datetime("%B", date(order_purchase_timestamp)) as order_month,
  count(o.order_id) as count_orders
from `sql_project.customers` c
join `sql_project.orders` o
on c.customer_id = o.customer_id
where order_status = "delivered"
group by customer_state,order_year,order_months,order_month
order by customer_state,order_year,order_months,order_month
),
table2 as
(
select *,
  lag(count_orders) over(order by customer_state,order_year,order_months) as p
revious_orders_count
from table1
order by customer_state,order_year,order_months
)
select *,
  count_orders-previous_orders_count as order_diff_month_by_month
from
  table2
```

Query Screenshot:



```
with table1 as
(
select
  customer_state,
  extract(year from order_purchase_timestamp) as order_year,
  extract(month from order_purchase_timestamp) as order_months,
  format_datetime("%B", date(order_purchase_timestamp)) as order_month,
  count(o.order_id) as count_orders
from `sql_project.customers` c
join `sql_project.orders` o
on c.customer_id = o.customer_id
where order_status = "delivered"
group by customer_state,order_year,order_months,order_month
order by customer_state,order_year,order_months,order_month
),
table2 as
(
select *,
  lag(count_orders) over(order by customer_state,order_year,order_months) as previous_orders_count
from table1
order by customer_state,order_year,order_months
)
select *,
  count_orders-previous_orders_count as order_diff_month_by_month
from
```

Result:



customer_state	order_year	order_months	order_month	count_orders	previous_orders_count	order_diff_month_by_month
AC	2017	1	January	2		
AC	2017	2	February	3	2	1
AC	2017	3	March	2	3	-1
AC	2017	4	April	5	2	3
AC	2017	5	May	8	5	3
AC	2017	6	June	4	8	-4
AC	2017	7	July	5	4	1
AC	2017	8	August	4	5	-1
AC	2017	9	September	5	4	1
AC	2017	10	October	5	5	0
AC	2017	11	November	5	5	0
AC	2017	12	December	5	5	0
AC	2018	1	January	6	5	1
AC	2018	2	February	3	6	-3

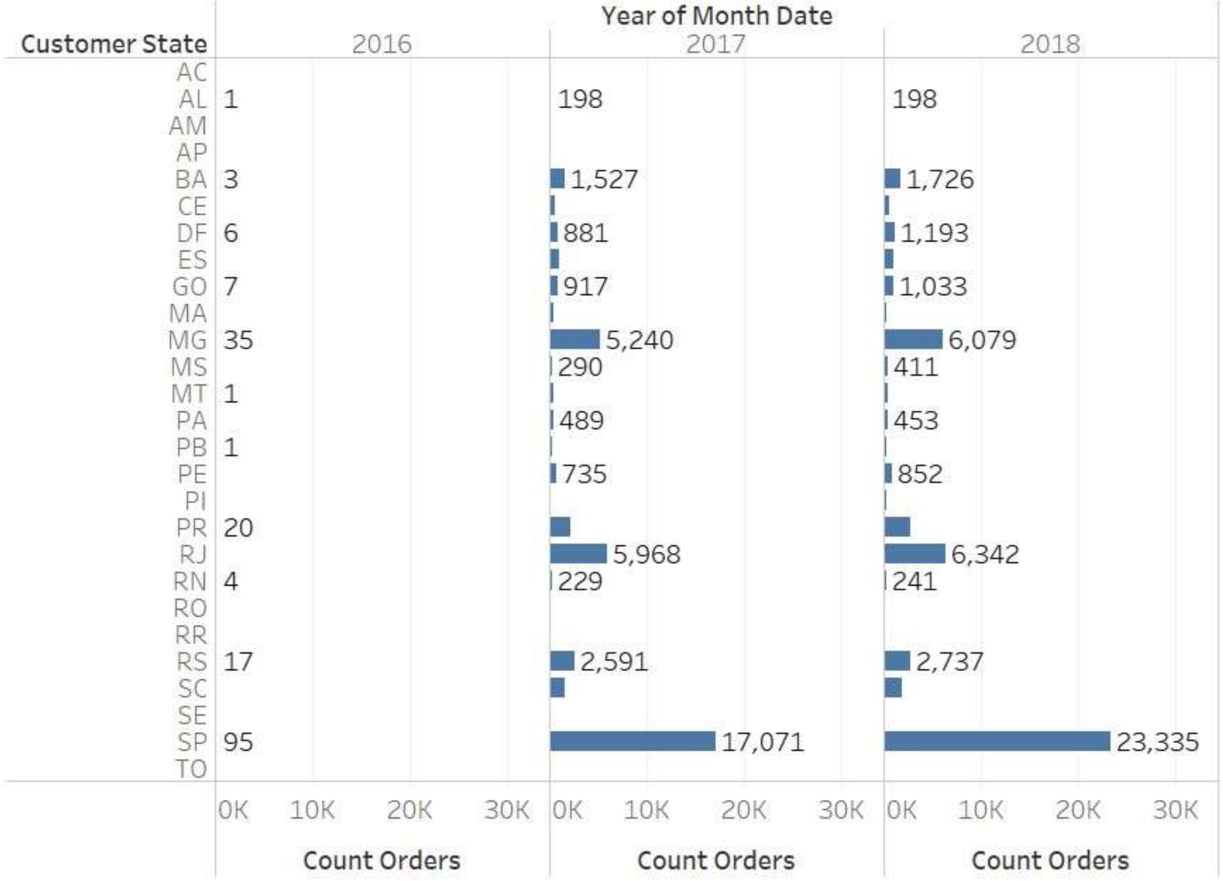
Insights:

Below two graphs are showing the analysis of count of orders for different states:

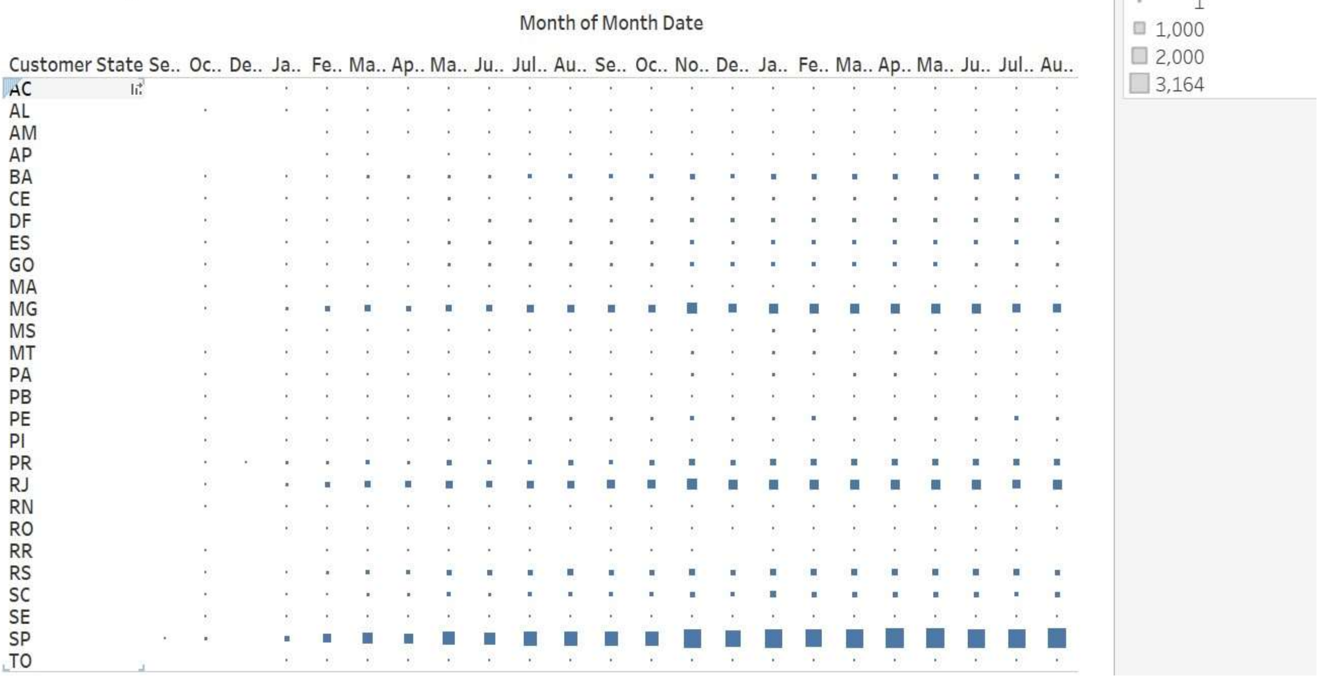
- Year-wise
- Month-wise

It can be observed that the orders have been gradually increasing with each passing month in almost all states. However, in some states, the orders have been placed rapidly in comparison with others, like, Sao Paulo, followed by Rio De Janeiro and Minas Gerais.

count of orders for different states in respective year



month by month count of orders for different states

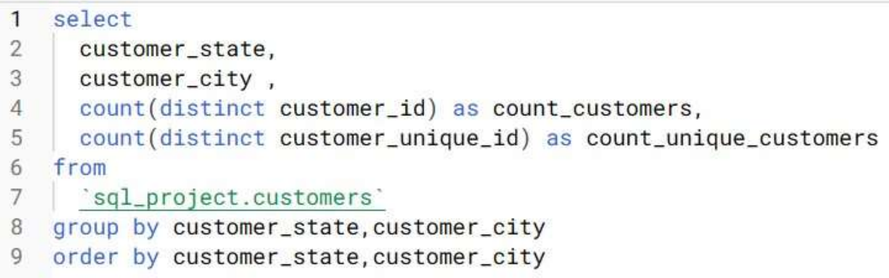


3.2. Distribution of customers across the states in Brazil

Query:

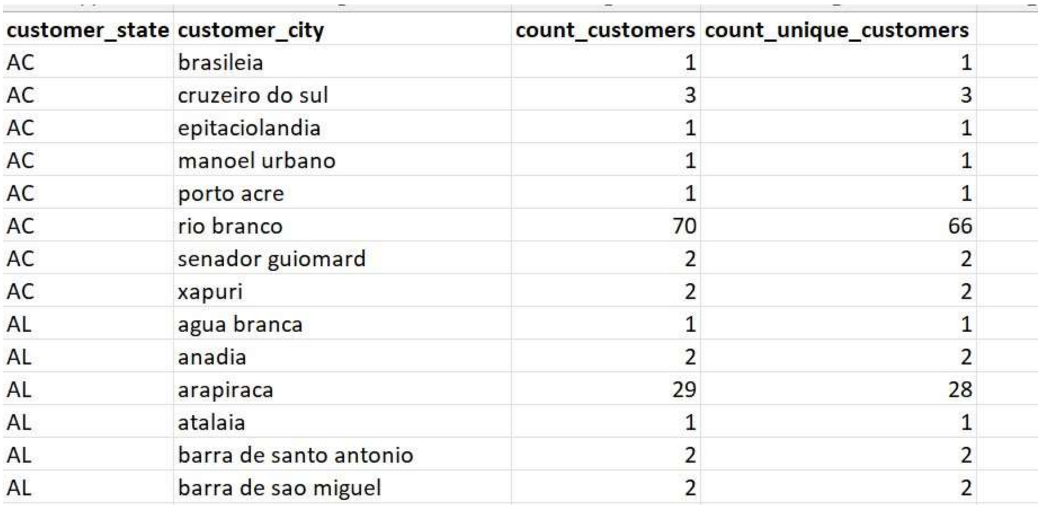
```
select
  customer_state,
  customer_city ,
  count(distinct customer_id) as count_customers,
  count(distinct customer_unique_id) as count_unique_customers
from
  `sql_project.customers`
group by customer_state,customer_city
order by customer_state,customer_city
```

Query Screenshot:



```
select
  customer_state,
  customer_city ,
  count(distinct customer_id) as count_customers,
  count(distinct customer_unique_id) as count_unique_customers
from
  `sql_project.customers`
group by customer_state,customer_city
order by customer_state,customer_city
```

Result:

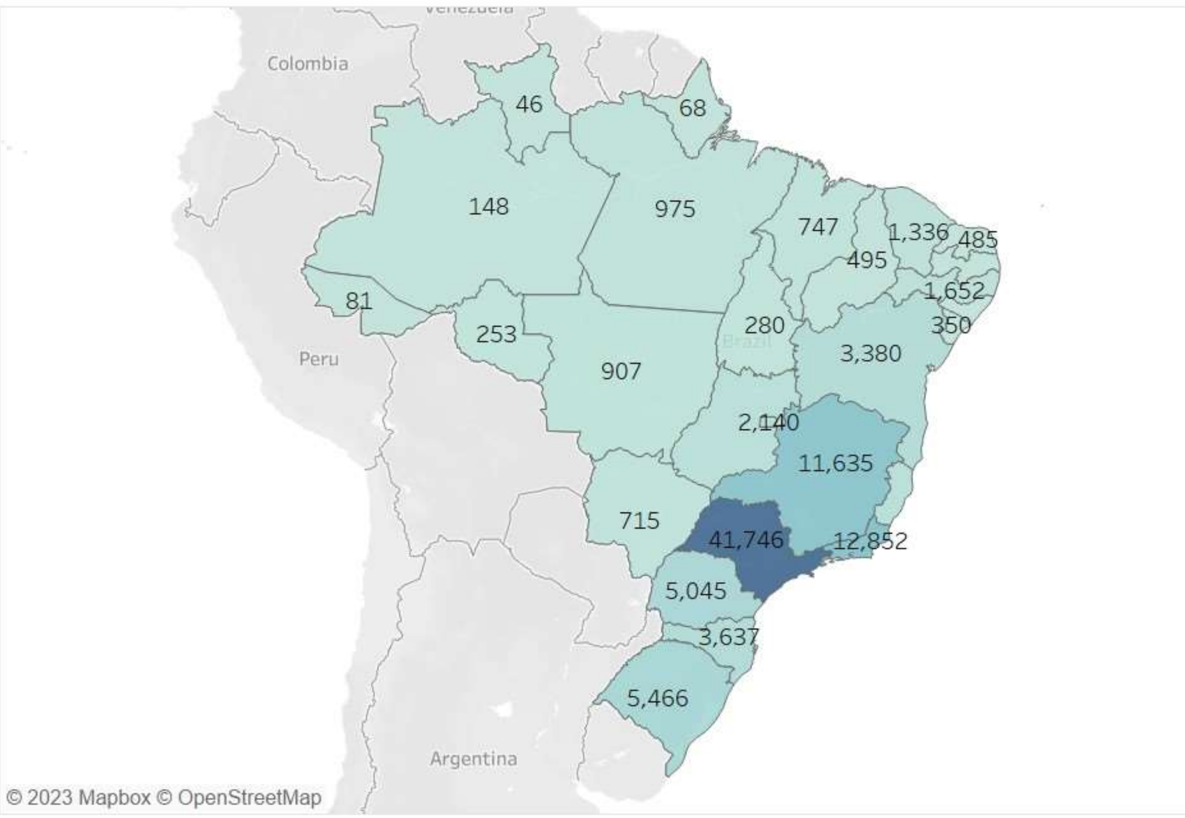


customer_state	customer_city	count_customers	count_unique_customers
AC	brasileia	1	1
AC	cruzeiro do sul	3	3
AC	epitaciolandia	1	1
AC	manoel urbano	1	1
AC	porto acre	1	1
AC	rio branco	70	66
AC	senador guiomard	2	2
AC	xapuri	2	2
AL	agua branca	1	1
AL	anadia	2	2
AL	arapiraca	29	28
AL	atalaia	1	1
AL	barra de santo antonio	2	2
AL	barra de sao miguel	2	2

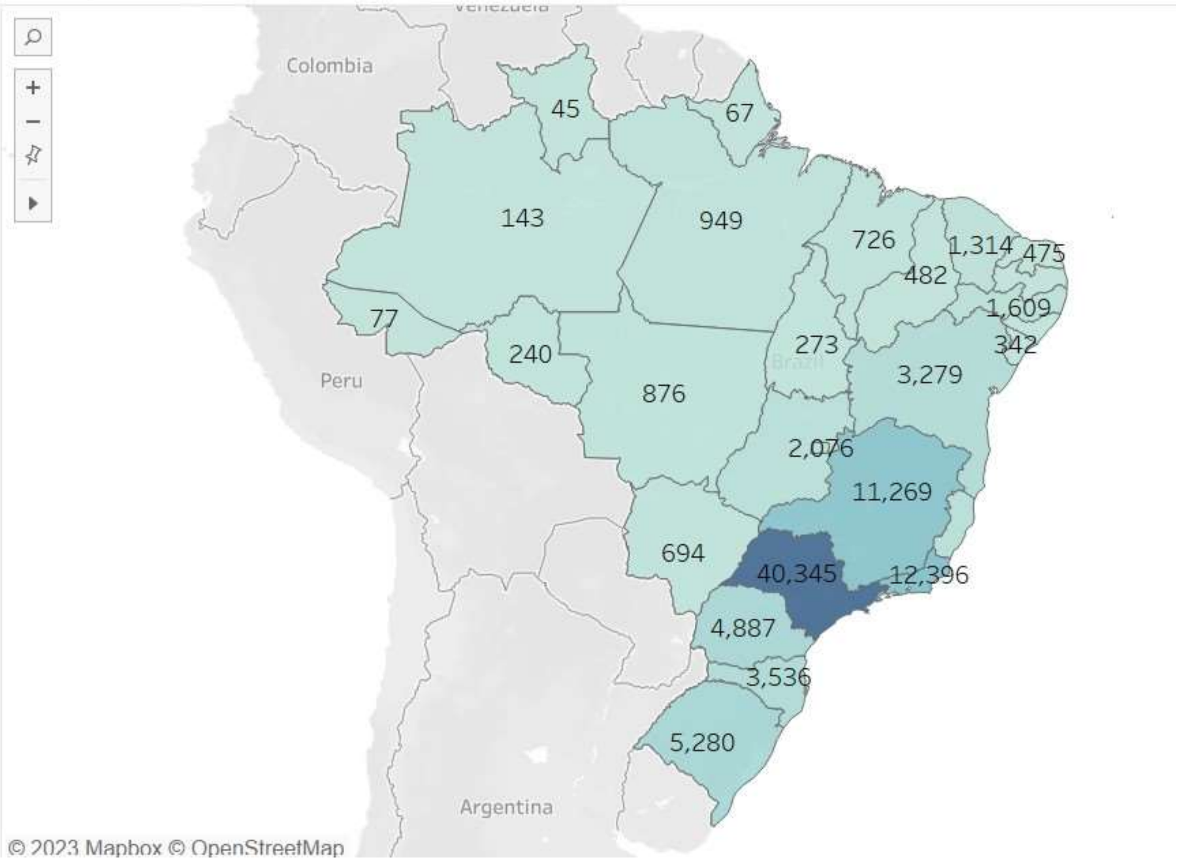
Insights:

The below graph is showing the distribution of customers and unique customers across all the states of Brazil. It can be observed clearly that maximum customers are from Sao Paulo.

Count of customers state-wise



unique customers count in states of Brazil



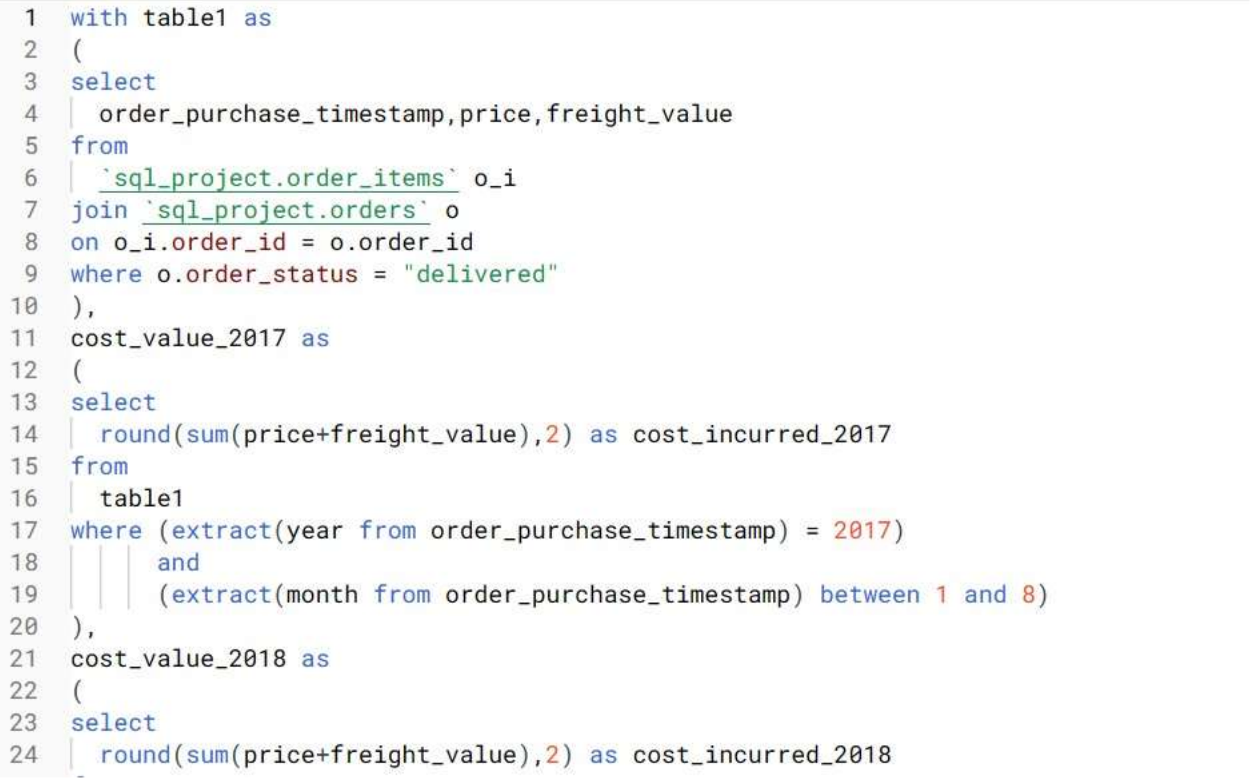
4. IMPACT ON ECONOMY: ANALYSE THE MONEY MOVEMENT BY E-COMMERCE BY LOOKING AT ORDER PRICES, FREIGHT AND OTHERS.

4.1. Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use “payment_value” column in payments table

Query:

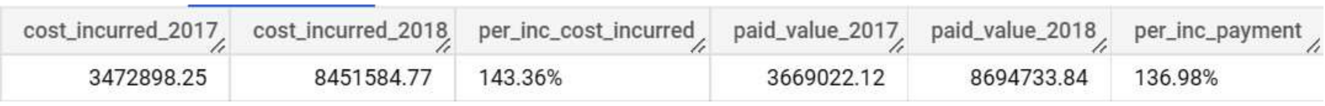
```
with table1 as
(
select
  order_purchase_timestamp,price,freight_value
from
  `sql_project.order_items` o_i
join `sql_project.orders` o
on o_i.order_id = o.order_id
where o.order_status = "delivered"
),
cost_value_2017 as
(
select
  round(sum(price+freight_value),2) as cost_incurred_2017
from
  table1
where (extract(year from order_purchase_timestamp) = 2017)
      and
      (extract(month from order_purchase_timestamp) between 1 and 8)
),
cost_value_2018 as
(
select
  round(sum(price+freight_value),2) as cost_incurred_2018
from
  table1
where (extract(year from order_purchase_timestamp) = 2018)
      and
      (extract(month from order_purchase_timestamp) between 1 and 8)
),
payment_value_2017 as
(
select round(sum(payment_value),2) as paid_value_2017
from sql_project.payments p
join `sql_project.orders` o
on p.order_id = o.order_id
where (extract(year from order_purchase_timestamp) = 2017)
      and
      (extract(month from order_purchase_timestamp) between 1 and 8)
),
payment_value_2018 as
(
select round(sum(payment_value),2) as paid_value_2018
from sql_project.payments p
join `sql_project.orders` o
on p.order_id = o.order_id
where (extract(year from order_purchase_timestamp) = 2018)
      and
      (extract(month from order_purchase_timestamp) between 1 and 8)
)
select
  cost_incurred_2017,
  cost_incurred_2018,
  concat(round(((cost_incurred_2018-
cost_incurred_2017)/cost_incurred_2017)*100,2),"%") as per_inc_cost_incurred,
  paid_value_2017,
  paid_value_2018,
  concat(round(((paid_value_2018-
paid_value_2017)/paid_value_2017)*100,2),"%") as per_inc_payment
from
  cost_value_2017, cost_value_2018, payment_value_2017, payment_value_2018
```

Query Screenshot:



```
with table1 as
(
select
  order_purchase_timestamp,price,freight_value
from
  `sql_project.order_items` o_i
join `sql_project.orders` o
on o_i.order_id = o.order_id
where o.order_status = "delivered"
),
cost_value_2017 as
(
select
  round(sum(price+freight_value),2) as cost_incurred_2017
from
  table1
where (extract(year from order_purchase_timestamp) = 2017)
      and
      (extract(month from order_purchase_timestamp) between 1 and 8)
),
cost_value_2018 as
(
select
  round(sum(price+freight_value),2) as cost_incurred_2018
```

Result:



cost_incurred_2017	cost_incurred_2018	per_inc_cost_incurred	paid_value_2017	paid_value_2018	per_inc_payment
3472898.25	8451584.77	143.36%	3669022.12	8694733.84	136.98%

Insights:

Cost incurred on any order includes the actual price of that order and the freight cost spent on that order. In 2017, the cost incurred is 3472898.25 and in 2018, the cost incurred is 8451584.77, therefore, the percent increase in cost incurred on orders from year 2017 to year 2018 is 143.36%.

The customer paid 3669022.12 and 8694733.84 on orders in year 2017 and 2018 respectively. Hence, the percent increase in payment orders is 136.98%.

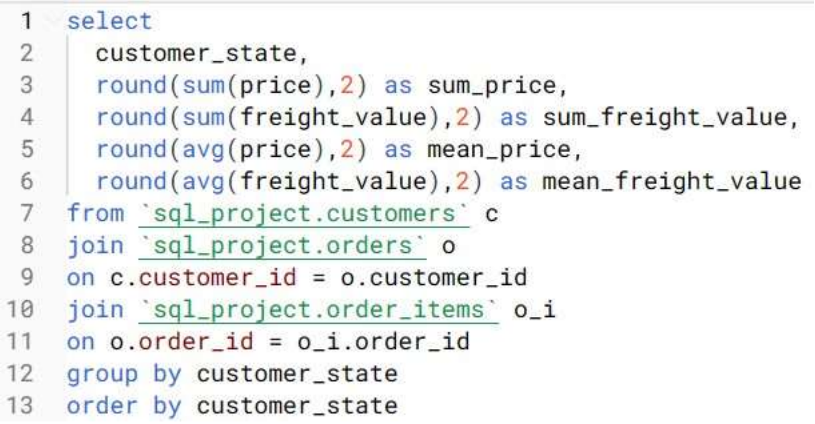
Therefore, percent increase in cost incurred on orders is quite greater than the percent increase in order payments by customers.

4.2. Mean & Sum of price and freight value by customer state

Query:

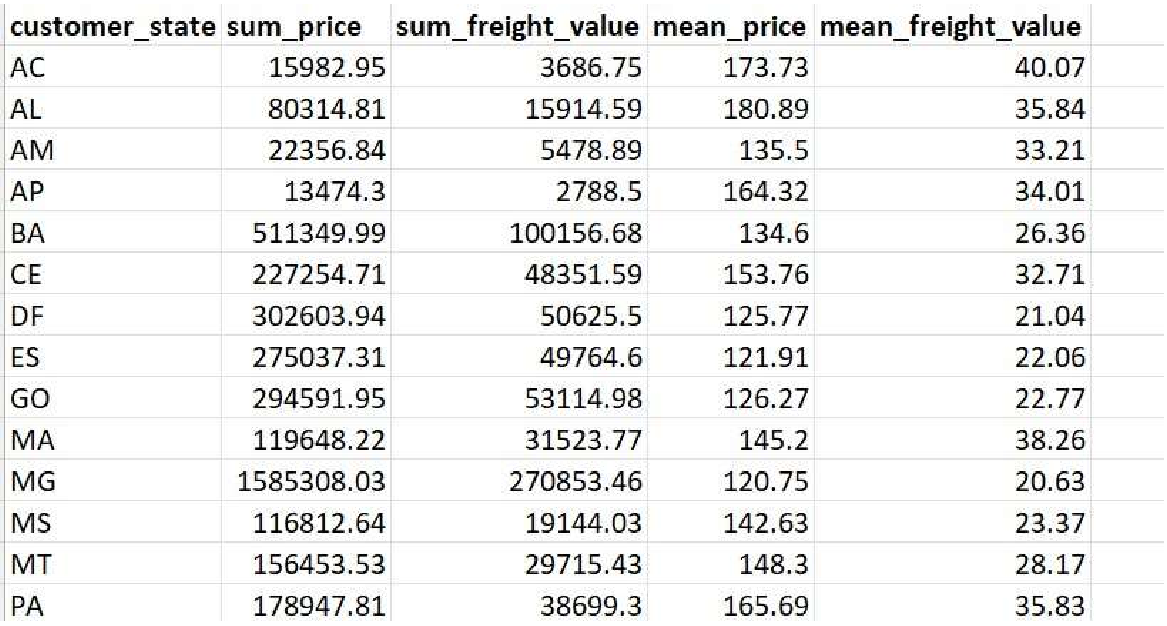
```
select
  customer_state,
  round(sum(price),2) as sum_price,
  round(sum(freight_value),2) as sum_freight_value,
  round(avg(price),2) as mean_price,
  round(avg(freight_value),2) as mean_freight_value
from `sql_project.customers` c
join `sql_project.orders` o
on c.customer_id = o.customer_id
join `sql_project.order_items` o_i
on o.order_id = o_i.order_id
group by customer_state
order by customer_state
```

Query Screenshot:



```
select
  customer_state,
  round(sum(price),2) as sum_price,
  round(sum(freight_value),2) as sum_freight_value,
  round(avg(price),2) as mean_price,
  round(avg(freight_value),2) as mean_freight_value
from `sql_project.customers` c
join `sql_project.orders` o
on c.customer_id = o.customer_id
join `sql_project.order_items` o_i
on o.order_id = o_i.order_id
group by customer_state
order by customer_state
```

Result:

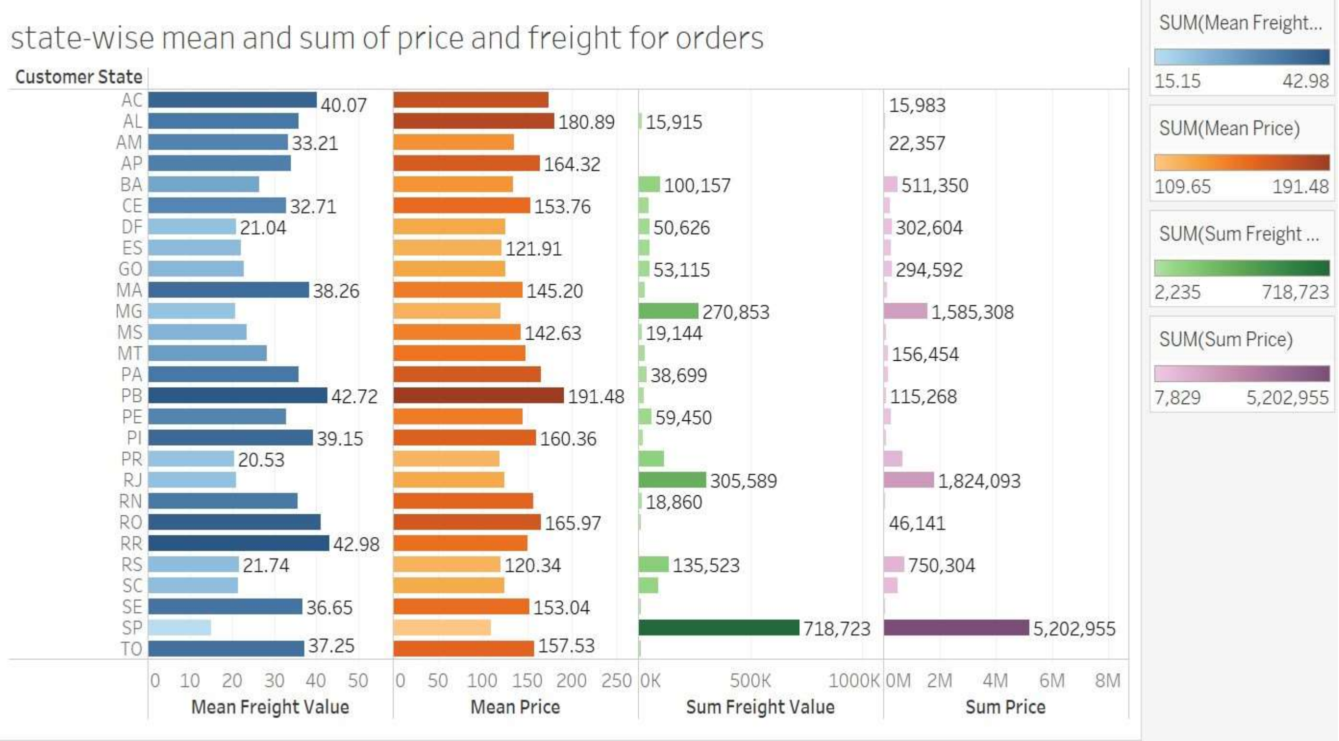


customer_state	sum_price	sum_freight_value	mean_price	mean_freight_value
AC	15982.95	3686.75	173.73	40.07
AL	80314.81	15914.59	180.89	35.84
AM	22356.84	5478.89	135.5	33.21
AP	13474.3	2788.5	164.32	34.01
BA	511349.99	100156.68	134.6	26.36
CE	227254.71	48351.59	153.76	32.71
DF	302603.94	50625.5	125.77	21.04
ES	275037.31	49764.6	121.91	22.06
GO	294591.95	53114.98	126.27	22.77
MA	119648.22	31523.77	145.2	38.26
MG	1585308.03	270853.46	120.75	20.63
MS	116812.64	19144.03	142.63	23.37
MT	156453.53	29715.43	148.3	28.17
PA	178947.81	38699.3	165.69	35.83

Insights:

The below chart shows the mean and sum of freight value and price value of all the orders across the states of Brazil.

It can be seen from the above chart that the maximum freight value of orders and maximum price of orders have been incurred on state Sao Paulo, followed by Rio De Janeiro.



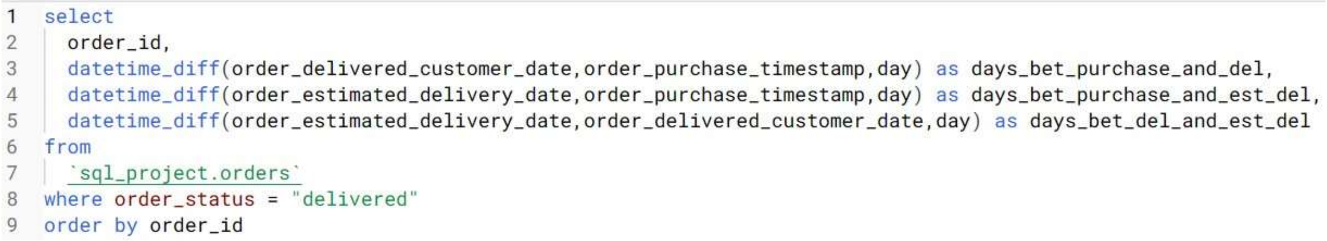
5. ANALYSIS ON SALES, FREIGHT AND DELIVERY TIME

5.1. Calculate days between purchasing, delivering and estimated delivery

Query:

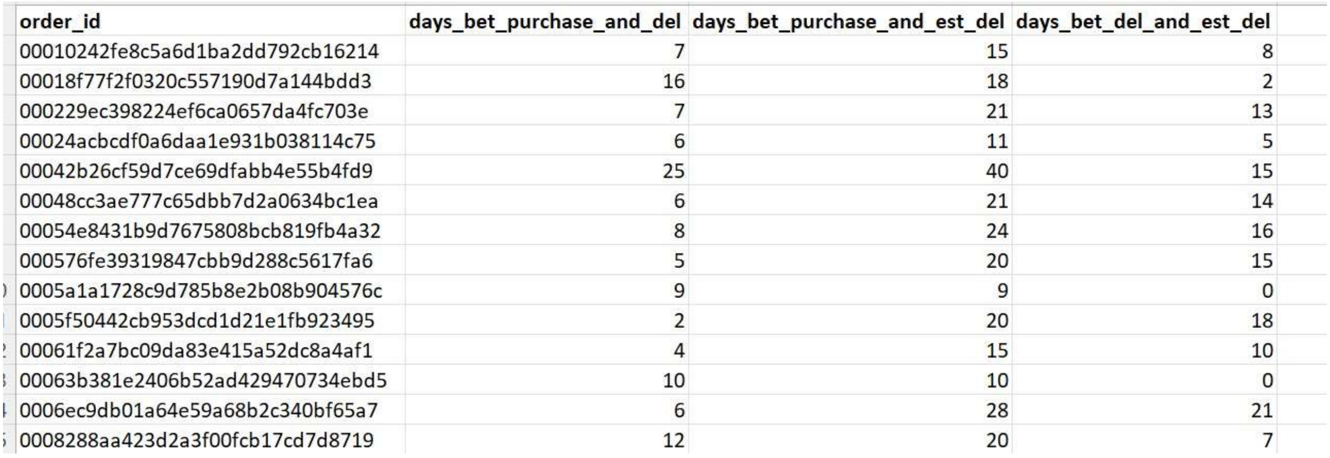
```
select
  order_id,
  datetime_diff(order_delivered_customer_date,order_purchase_timestamp,day) as
 days_bet_purchase_and_del,
  datetime_diff(order_estimated_delivery_date,order_purchase_timestamp,day) as
 days_bet_purchase_and_est_del,
  datetime_diff(order_estimated_delivery_date,order_delivered_customer_date,da
y) as days_bet_del_and_est_del
from
  `sql_project.orders`
where order_status = "delivered"
order by order_id
```

Query Screenshot:



```
select
  order_id,
  datetime_diff(order_delivered_customer_date,order_purchase_timestamp,day) as days_bet_purchase_and_del,
  datetime_diff(order_estimated_delivery_date,order_purchase_timestamp,day) as days_bet_purchase_and_est_del,
  datetime_diff(order_estimated_delivery_date,order_delivered_customer_date,day) as days_bet_del_and_est_del
from
  `sql_project.orders`
where order_status = "delivered"
order by order_id
```

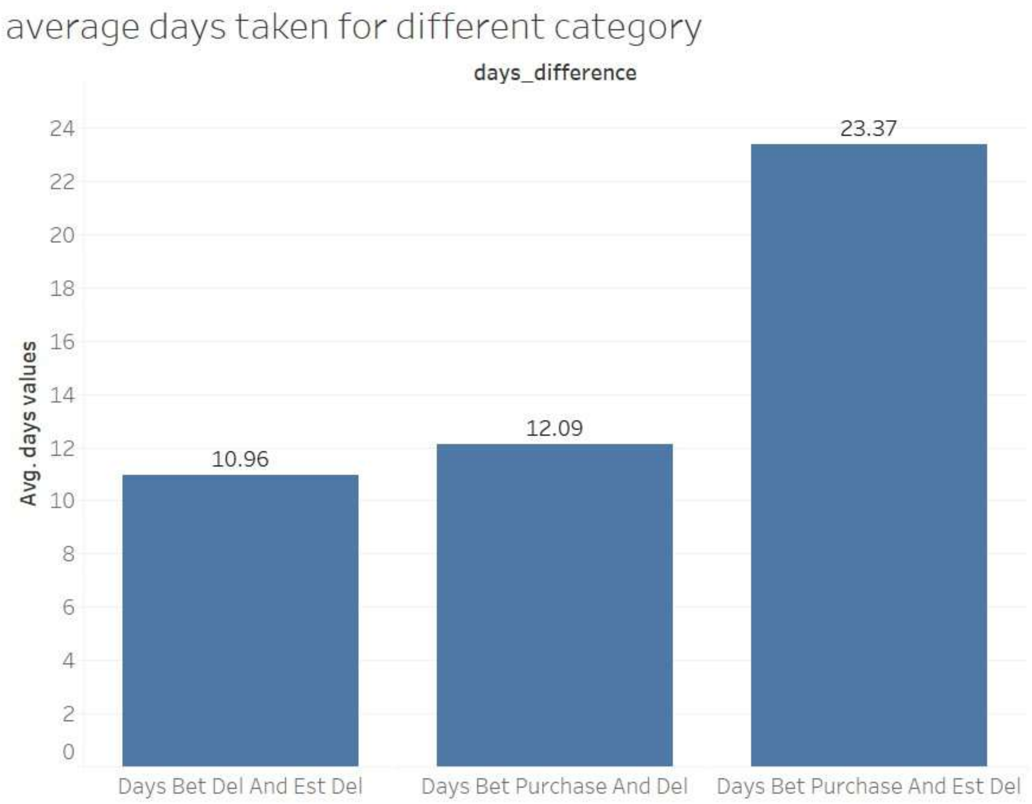
Result:



order_id	days_bet_purchase_and_del	days_bet_purchase_and_est_del	days_bet_del_and_est_del
00010242fe8c5a6d1ba2dd792cb16214	7	15	8
00018f77f2f0320c557190d7a144bdd3	16	18	2
000229ec398224ef6ca0657da4fc703e	7	21	13
00024acbcdf0a6daa1e931b038114c75	6	11	5
00042b26cf59d7ce69dfabb4e55b4fd9	25	40	15
00048cc3ae777c65dbb7d2a0634bc1ea	6	21	14
00054e8431b9d7675808bcb819fb4a32	8	24	16
000576fe39319847cbb9d288c5617fa6	5	20	15
0005a1a1728c9d785b8e2b08b904576c	9	9	0
0005f50442cb953dcd1d21e1fb923495	2	20	18
00061f2a7bc09da83e415a52dc8a4af1	4	15	10
00063b381e2406b52ad429470734ebd5	10	10	0
0006ec9db01a64e59a68b2c340bf65a7	6	28	21
0008288aa423d2a3f00fcb17cd7d8719	12	20	7

Insights:

Below chart shows the average days taken for all categories of delivery.
The average days taken between delivering and estimated delivery is 10.96 days.
The average days taken between purchasing and delivering is 12.09 days.
The average days taken between purchasing and estimated delivery is 23.37 days.



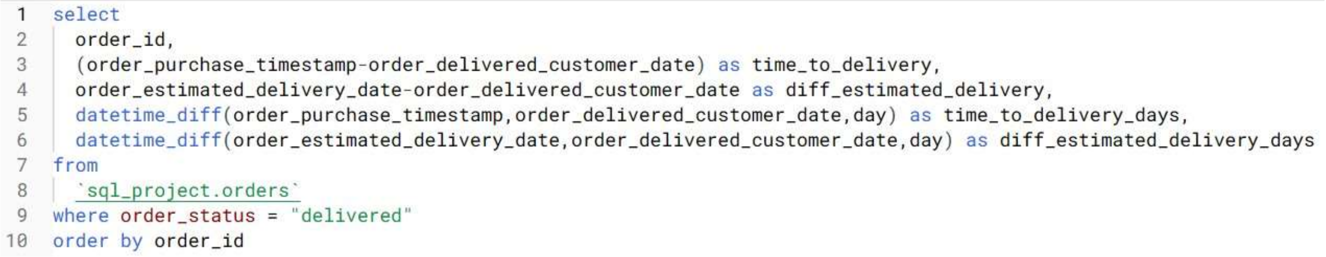
5.2. Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:

- **time_to_delivery = order_purchase_timestamp-order_delivered_customer_date**
- **diff_estimated_delivery=order_estimated_delivery_date-order_delivered_customer_date**

Query:

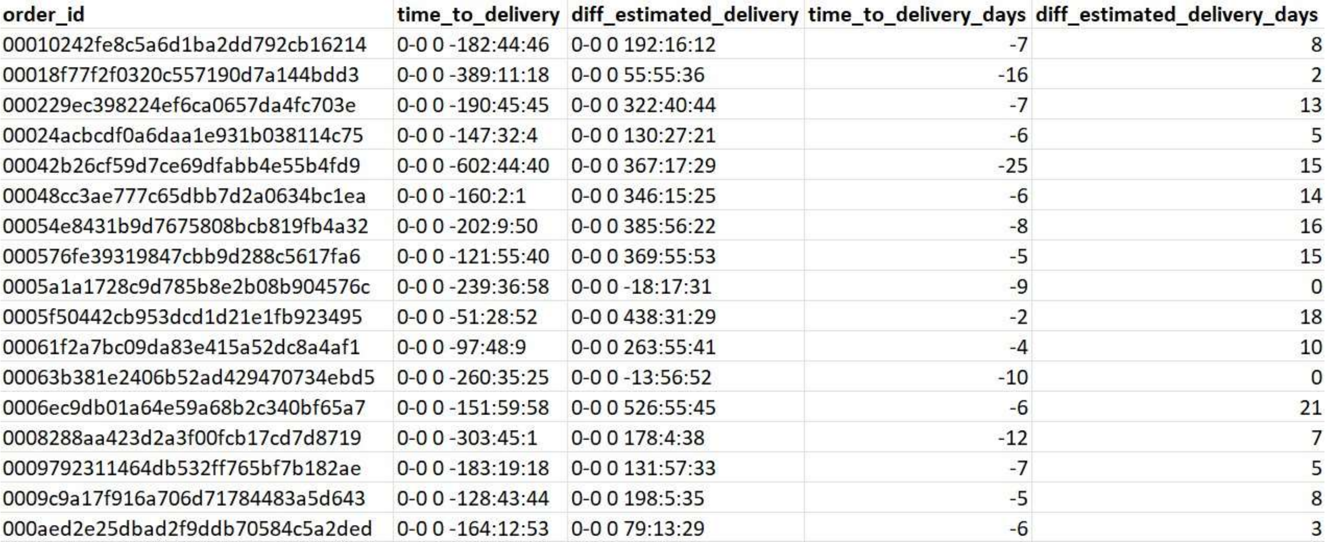
```
select
  order_id,
  (order_purchase_timestamp-
order_delivered_customer_date) as time_to_delivery,
  order_estimated_delivery_date-
order_delivered_customer_date as diff_estimated_delivery,
  datetime_diff(order_purchase_timestamp,order_delivered_customer_date,day) as
 time_to_delivery_days,
  datetime_diff(order_estimated_delivery_date,order_delivered_customer_date,da
y) as diff_estimated_delivery_days
from
  `sql_project.orders`
where order_status = "delivered"
order by order_id
```

Query Screenshot:



```
select
  order_id,
  (order_purchase_timestamp-order_delivered_customer_date) as time_to_delivery,
  order_estimated_delivery_date-order_delivered_customer_date as diff_estimated_delivery,
  datetime_diff(order_purchase_timestamp,order_delivered_customer_date,day) as time_to_delivery_days,
  datetime_diff(order_estimated_delivery_date,order_delivered_customer_date,day) as diff_estimated_delivery_days
from
  `sql_project.orders`
where order_status = "delivered"
order by order_id
```

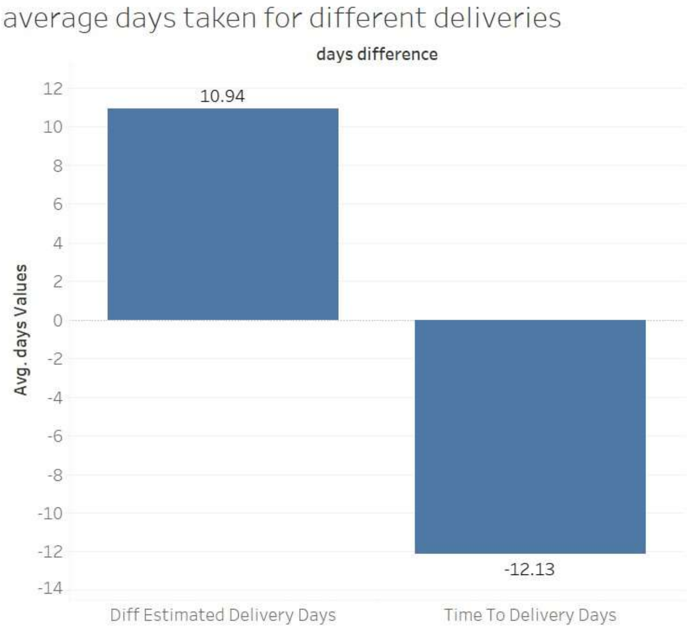
Result:



order_id	time_to_delivery	diff_estimated_delivery	time_to_delivery_days	diff_estimated_delivery_days
00010242fe8c5a6d1ba2dd792cb16214	0-0 0 -182:44:46	0-0 0 192:16:12	-7	8
00018f77f2f0320c557190d7a144bdd3	0-0 0 -389:11:18	0-0 0 55:55:36	-16	2
000229ec398224ef6ca0657da4fc703e	0-0 0 -190:45:45	0-0 0 322:40:44	-7	13
00024acbcdf0a6daa1e931b038114c75	0-0 0 -147:32:4	0-0 0 130:27:21	-6	5
00042b26cf59d7ce69dfabb4e55b4fd9	0-0 0 -602:44:40	0-0 0 367:17:29	-25	15
00048cc3ae777c65dbb7d2a0634bc1ea	0-0 0 -160:2:1	0-0 0 346:15:25	-6	14
00054e8431b9d7675808bcb819fb4a32	0-0 0 -202:9:50	0-0 0 385:56:22	-8	16
000576fe39319847cbb9d288c5617fa6	0-0 0 -121:55:40	0-0 0 369:55:53	-5	15
0005a1a1728c9d785b8e2b08b904576c	0-0 0 -239:36:58	0-0 0 -18:17:31	-9	0
0005f50442cb953dcd1d21e1fb923495	0-0 0 -51:28:52	0-0 0 438:31:29	-2	18
00061f2a7bc09da83e415a52dc8a4af1	0-0 0 -97:48:9	0-0 0 263:55:41	-4	10
00063b381e2406b52ad429470734ebd5	0-0 0 -260:35:25	0-0 0 -13:56:52	-10	0
0006ec9db01a64e59a68b2c340bf65a7	0-0 0 -151:59:58	0-0 0 526:55:45	-6	21
0008288aa423d2a3f00fcb17cd7d8719	0-0 0 -303:45:1	0-0 0 178:4:38	-12	7
0009792311464db532ff765bf7b182ae	0-0 0 -183:19:18	0-0 0 131:57:33	-7	5
0009c9a17f916a706d71784483a5d643	0-0 0 -128:43:44	0-0 0 198:5:35	-5	8
000aed2e25dbad2f9ddb70584c5a2ded	0-0 0 -164:12:53	0-0 0 79:13:29	-6	3

Insights:

The average days taken between delivering and estimated delivery is 10.94 days.
The average days taken between purchasing and delivering is 12.13 days.
Please note: days have been considered as absolute value.

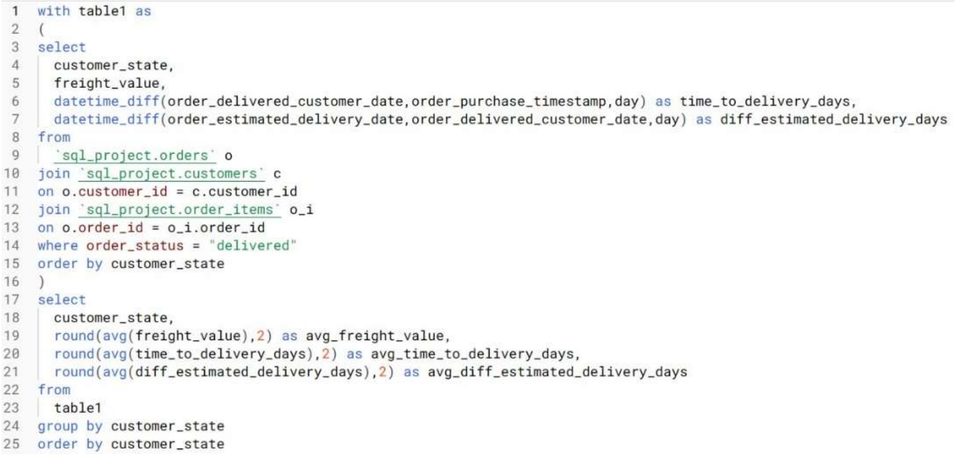


5.3. Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

Query:

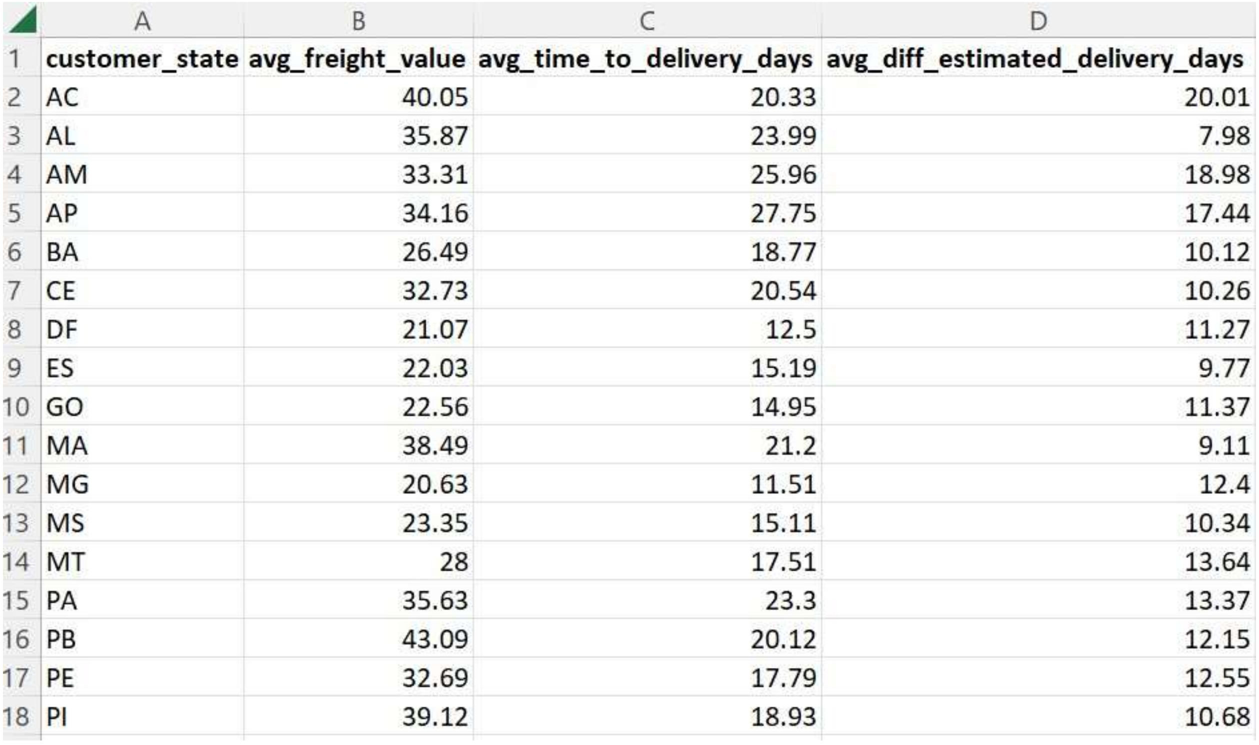
```
with table1 as
(
select
  customer_state,
  freight_value,
  datetime_diff(order_delivered_customer_date,order_purchase_timestamp,day) as
 time_to_delivery_days,
  datetime_diff(order_estimated_delivery_date,order_delivered_customer_date,da
y) as diff_estimated_delivery_days
from
  `sql_project.orders` o
join `sql_project.customers` c
on o.customer_id = c.customer_id
join `sql_project.order_items` o_i
on o.order_id = o_i.order_id
where order_status = "delivered"
order by customer_state
)
select
  customer_state,
  round(avg(freight_value),2) as avg_freight_value,
  round(avg(time_to_delivery_days),2) as avg_time_to_delivery_days,
  round(avg(diff_estimated_delivery_days),2) as avg_diff_estimated_delivery_da
ys
from
  table1
group by customer_state
order by customer_state
```

Query Screenshot:



```
with table1 as
(
select
  customer_state,
  freight_value,
  datetime_diff(order_delivered_customer_date,order_purchase_timestamp,day) as time_to_delivery_days,
  datetime_diff(order_estimated_delivery_date,order_delivered_customer_date,day) as diff_estimated_delivery_days
from
  `sql_project.orders` o
join `sql_project.customers` c
on o.customer_id = c.customer_id
join `sql_project.order_items` o_i
on o.order_id = o_i.order_id
where order_status = "delivered"
order by customer_state
)
select
  customer_state,
  round(avg(freight_value),2) as avg_freight_value,
  round(avg(time_to_delivery_days),2) as avg_time_to_delivery_days,
  round(avg(diff_estimated_delivery_days),2) as avg_diff_estimated_delivery_days
from
  table1
group by customer_state
order by customer_state
```

Result:



	A	B	C	D
1	customer_state	avg_freight_value	avg_time_to_delivery_days	avg_diff_estimated_delivery_days
2	AC	40.05	20.33	20.01
3	AL	35.87	23.99	7.98
4	AM	33.31	25.96	18.98
5	AP	34.16	27.75	17.44
6	BA	26.49	18.77	10.12
7	CE	32.73	20.54	10.26
8	DF	21.07	12.5	11.27
9	ES	22.03	15.19	9.77
10	GO	22.56	14.95	11.37
11	MA	38.49	21.2	9.11
12	MG	20.63	11.51	12.4
13	MS	23.35	15.11	10.34
14	MT	28	17.51	13.64
15	PA	35.63	23.3	13.37
16	PB	43.09	20.12	12.15
17	PE	32.69	17.79	12.55
18	PI	39.12	18.93	10.68

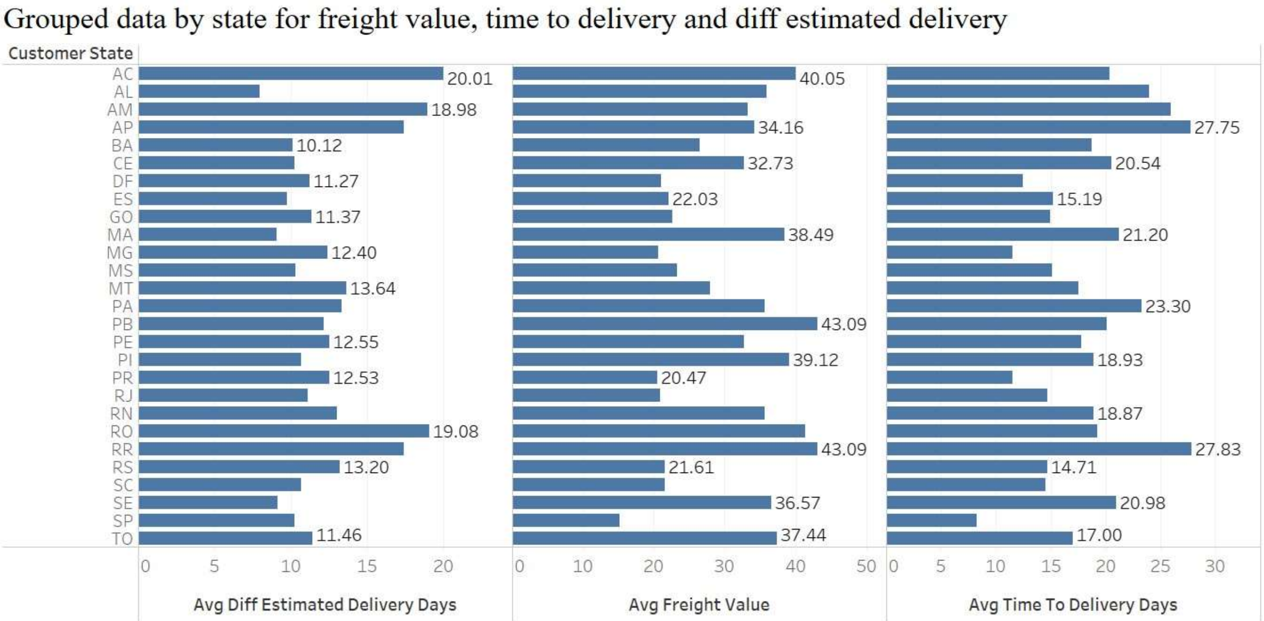
Insights:

Below chart shows the freight value, time to delivery and difference estimated delivery in all the states.

State *AC* has the maximum average difference estimated delivery days.

State *PB* and *RR* has the maximum average freight value.

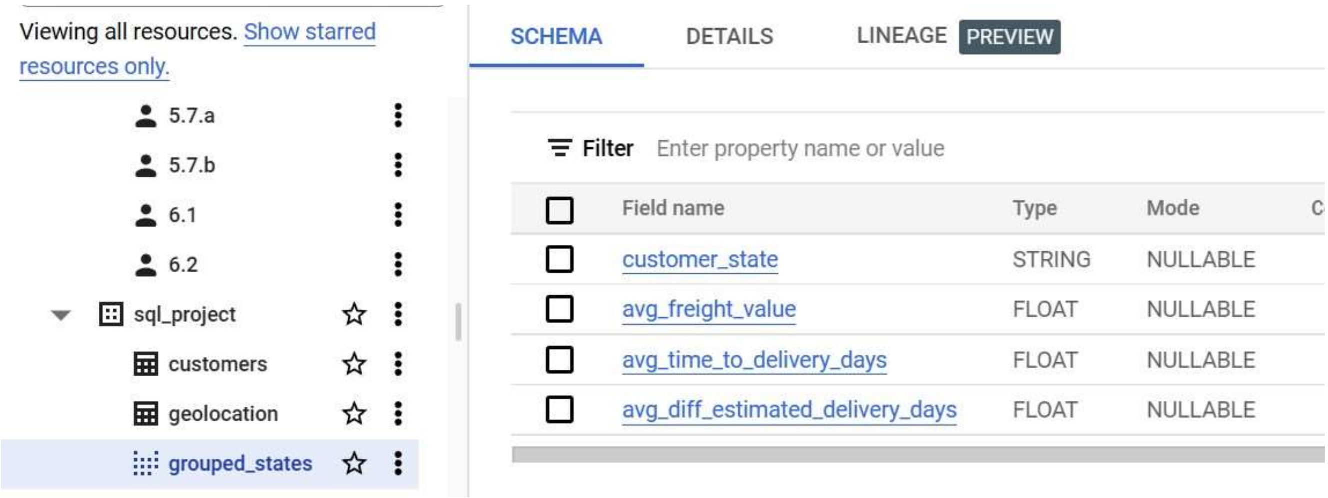
State *RR* has the maximum average time to delivery days.



5.4. Sort the data to get the following:

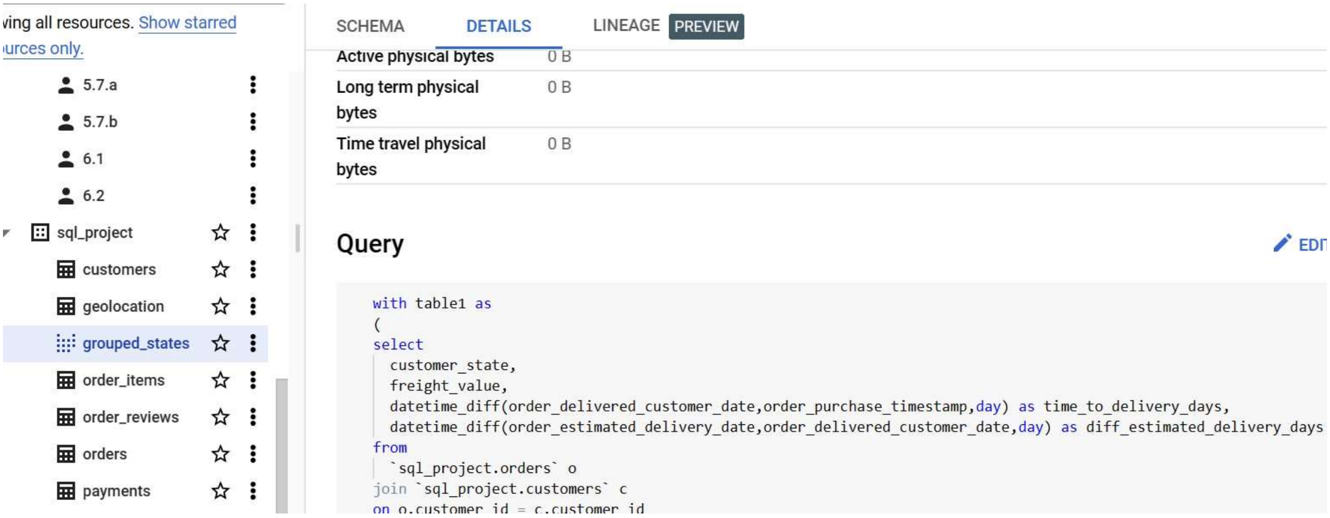
View has been created for the next three problems.

Schema screenshot:



☐	Field name	Type	Mode	C
☐	customer_state	STRING	NULLABLE	
☐	avg_freight_value	FLOAT	NULLABLE	
☐	avg_time_to_delivery_days	FLOAT	NULLABLE	
☐	avg_diff_estimated_delivery_days	FLOAT	NULLABLE	

Query detail screenshot:



Active physical bytes 0 B

Long term physical bytes 0 B

Time travel physical bytes 0 B

Query

```
with table1 as
(
select
  customer_state,
  freight_value,
  datetime_diff(order_delivered_customer_date,order_purchase_timestamp,day) as time_to_delivery_days,
  datetime_diff(order_estimated_delivery_date,order_delivered_customer_date,day) as diff_estimated_delivery_days
from
  `sql_project.orders` o
join `sql_project.customers` c
on o.customer_id = c.customer_id
```

Query of view "grouped_states":

```
with table1 as
(
select
  customer_state,
  freight_value,
  datetime_diff(order_delivered_customer_date,order_purchase_timestamp,day) as time_to_delivery_days,
  datetime_diff(order_estimated_delivery_date,order_delivered_customer_date,day) as diff_estimated_delivery_days
from
```

```
  `sql_project.orders` o
join `sql_project.customers` c
on o.customer_id = c.customer_id
join `sql_project.order_items` o_i
on o.order_id = o_i.order_id
where order_status = "delivered"
order by customer_state
)
select
  customer_state,
  round(avg(freight_value),2) as avg_freight_value,
  round(avg(time_to_delivery_days),2) as avg_time_to_delivery_days,
  round(avg(diff_estimated_delivery_days),2) as avg_diff_estimated_delivery_days
from
  table1
group by customer_state
order by customer_state
```

5.5. Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

5.5.a. Top 5 states with highest average freight value

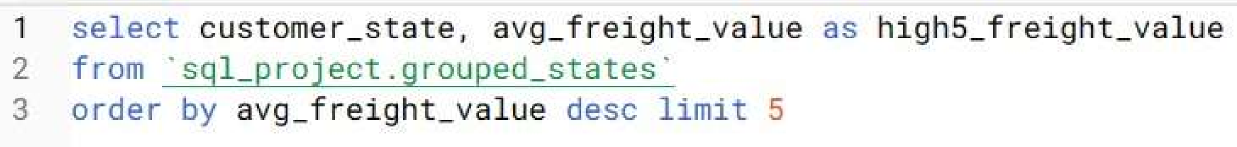
5.5.b. Top 5 states with lowest average freight value

Query:

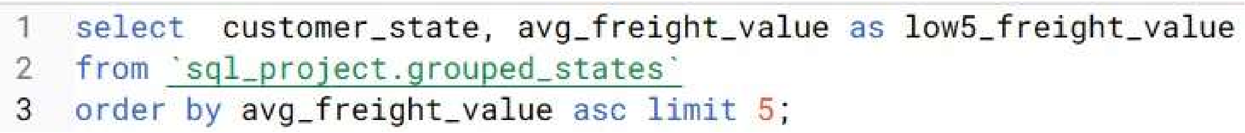
```
select customer_state, avg_freight_value as high5_freight_value
from `sql_project.grouped_states`
order by avg_freight_value desc limit 5;

select  customer_state, avg_freight_value as low5_freight_value
from `sql_project.grouped_states`
order by avg_freight_value asc limit 5;
```

Query Screenshot:

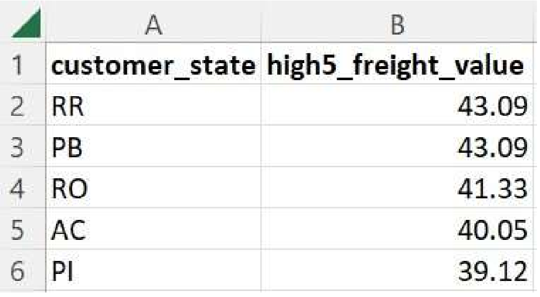


```
select customer_state, avg_freight_value as high5_freight_value
from `sql_project.grouped_states`
order by avg_freight_value desc limit 5
```

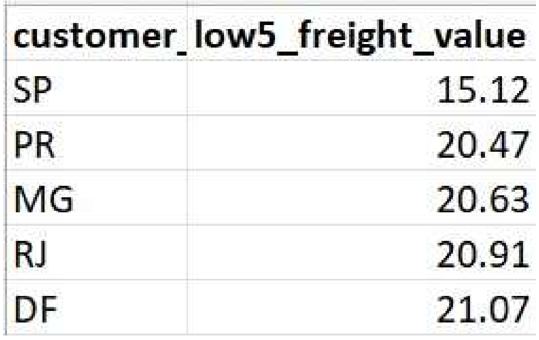


```
select  customer_state, avg_freight_value as low5_freight_value
from `sql_project.grouped_states`
order by avg_freight_value asc limit 5;
```

Result:



	A	B
1	customer_state	high5_freight_value
2	RR	43.09
3	PB	43.09
4	RO	41.33
5	AC	40.05
6	PI	39.12

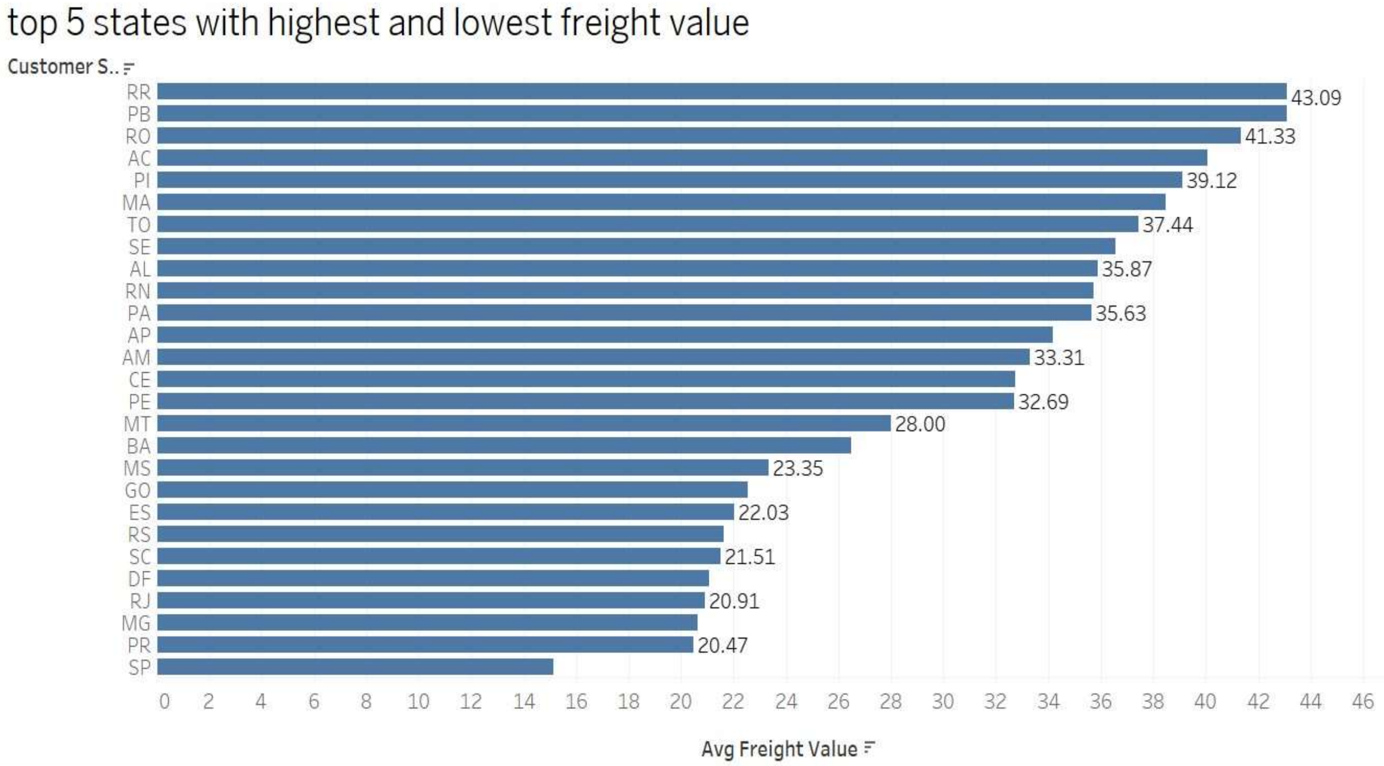


customer_	low5_freight_value
SP	15.12
PR	20.47
MG	20.63
RJ	20.91
DF	21.07

Insights:

Top 5 states with highest freight value are the top five states mentioned in graph, that are, *RR,PB,RO,AC and PI.*

Top 5 states with lowest freight value are the below five states mentioned in graph that are, *SP,PR,MG,RJ and DF.*



5.6. Top 5 states with highest/lowest average time to delivery

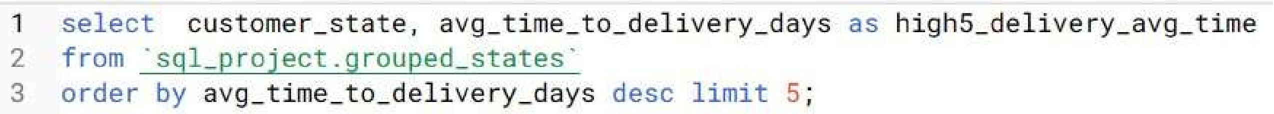
5.6.a. Top 5 states with highest average time to delivery
5.6.b. Top 5 states with lowest average time to delivery

Query:

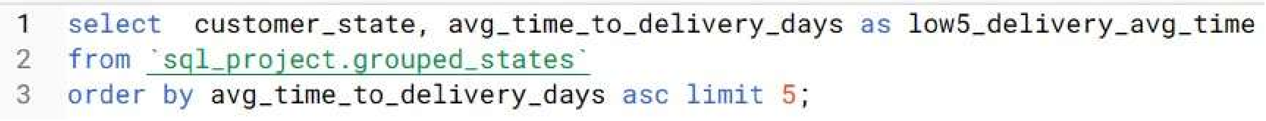
```
select
      customer_state,
      avg_time_to_delivery_days as high5_delivery_avg_time
from `sql_project.grouped_states`
order by avg_time_to_delivery_days desc limit 5;

select
      customer_state,
      avg_time_to_delivery_days as low5_delivery_avg_time
from `sql_project.grouped_states`
order by avg_time_to_delivery_days asc limit 5;
```

Query Screenshot:

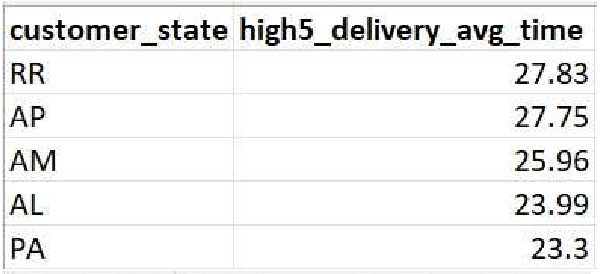


```
select  customer_state, avg_time_to_delivery_days as high5_delivery_avg_time
from `sql_project.grouped_states`
order by avg_time_to_delivery_days desc limit 5;
```

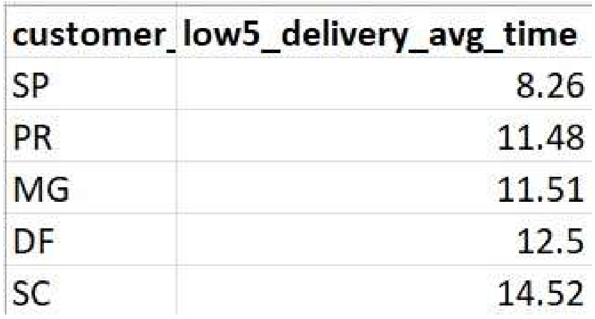


```
select  customer_state, avg_time_to_delivery_days as low5_delivery_avg_time
from `sql_project.grouped_states`
order by avg_time_to_delivery_days asc limit 5;
```

Result:



customer_state	high5_delivery_avg_time
RR	27.83
AP	27.75
AM	25.96
AL	23.99
PA	23.3

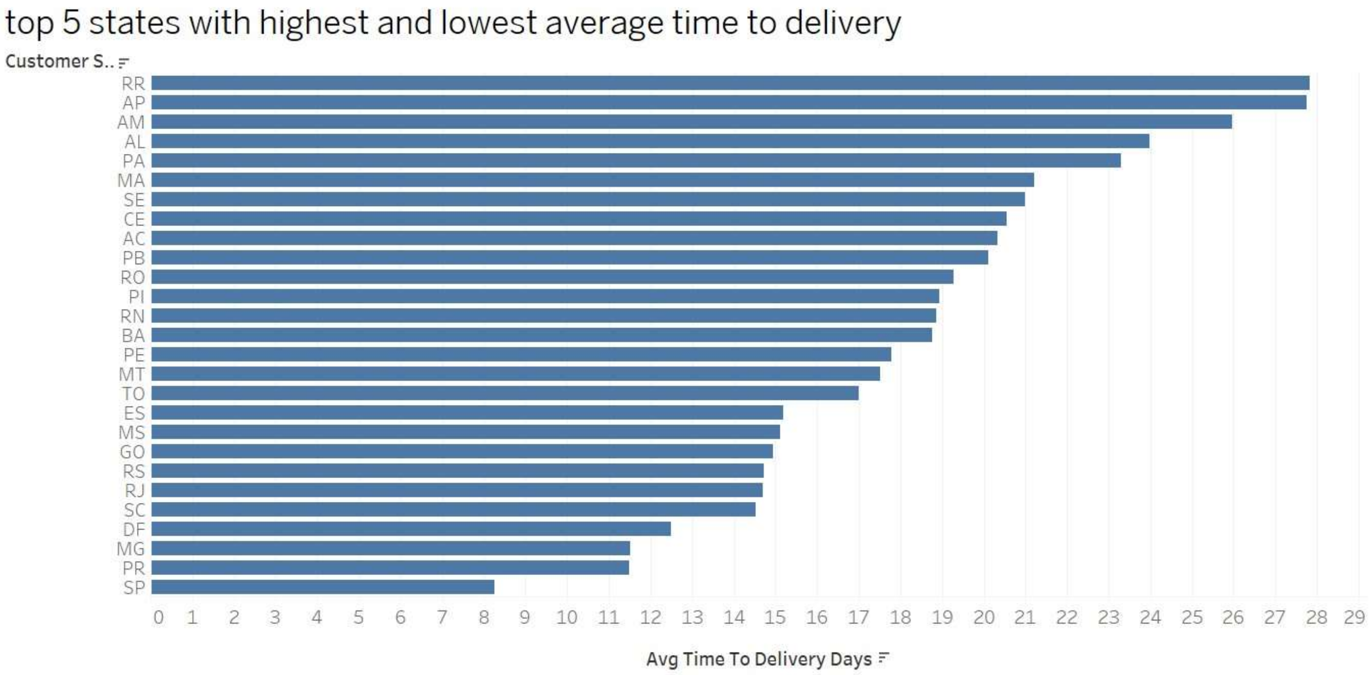


customer_	low5_delivery_avg_time
SP	8.26
PR	11.48
MG	11.51
DF	12.5
SC	14.52

Insights:

Top 5 states with highest average time to delivery are the top five states mentioned in graph, that are, *RR,AP,AM,AL and PA.*

Top 5 states with lowest average time to delivery are the below five states mentioned in graph that are, *SP,PR,MG,DF and SC.*



5.7. Top 5 states where delivery is really fast/ not so fast compared to estimated date

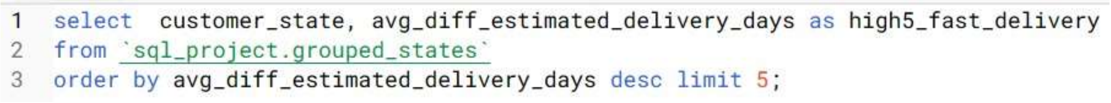
5.7.a. Top 5 states where delivery is really fast compared to estimated date
5.7.b. Top 5 states where delivery is not so fast compared to estimated date

Query:

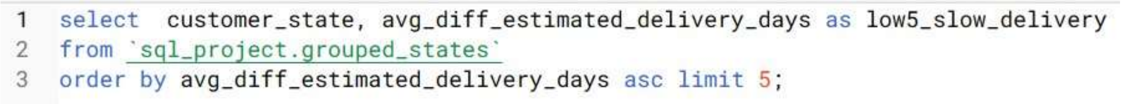
```
select
  customer_state,
  avg_diff_estimated_delivery_days as high5_fast_delivery
from `sql_project.grouped_states`
order by avg_diff_estimated_delivery_days desc limit 5;

select
  customer_state,
  avg_diff_estimated_delivery_days as low5_slow_delivery
from `sql_project.grouped_states`
order by avg_diff_estimated_delivery_days asc limit 5;
```

Query Screenshot:

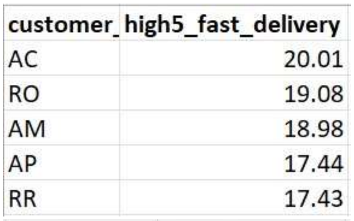


```
select  customer_state, avg_diff_estimated_delivery_days as high5_fast_delivery
from `sql_project.grouped_states`
order by avg_diff_estimated_delivery_days desc limit 5;
```

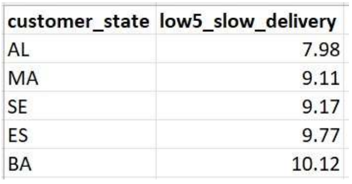


```
select  customer_state, avg_diff_estimated_delivery_days as low5_slow_delivery
from `sql_project.grouped_states`
order by avg_diff_estimated_delivery_days asc limit 5;
```

Result:



customer_	high5_fast_delivery
AC	20.01
RO	19.08
AM	18.98
AP	17.44
RR	17.43

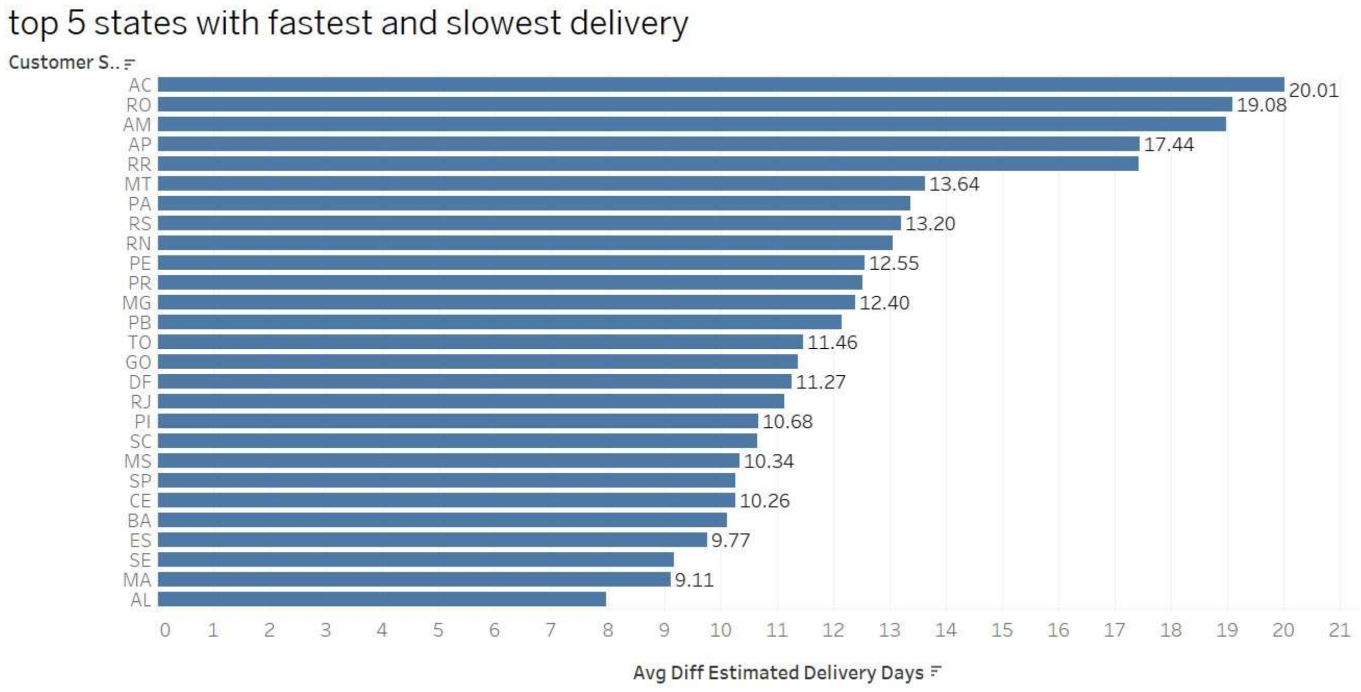


customer_state	low5_slow_delivery
AL	7.98
MA	9.11
SE	9.17
ES	9.77
BA	10.12

Insights:

Top 5 states with fastest delivery are the top five states mentioned in graph, that are, *AC,RO,AM,AP, and RR.*

Top 5 states with slowest delivery are the below five states mentioned in graph that are, *AL,MA,SE,ES and BA.*



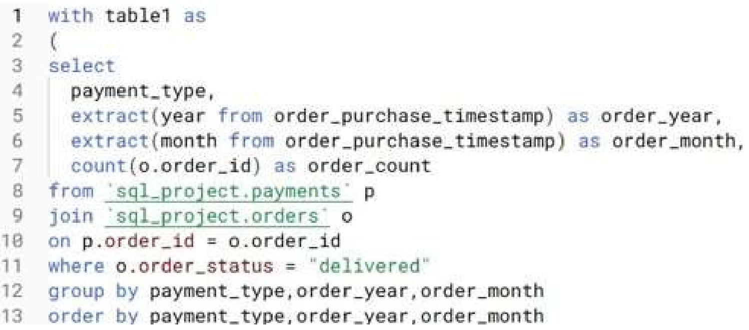
6. PAYMENT TYPE ANALYSIS:

6.1. Month over Month count of orders for different payment types

Query:

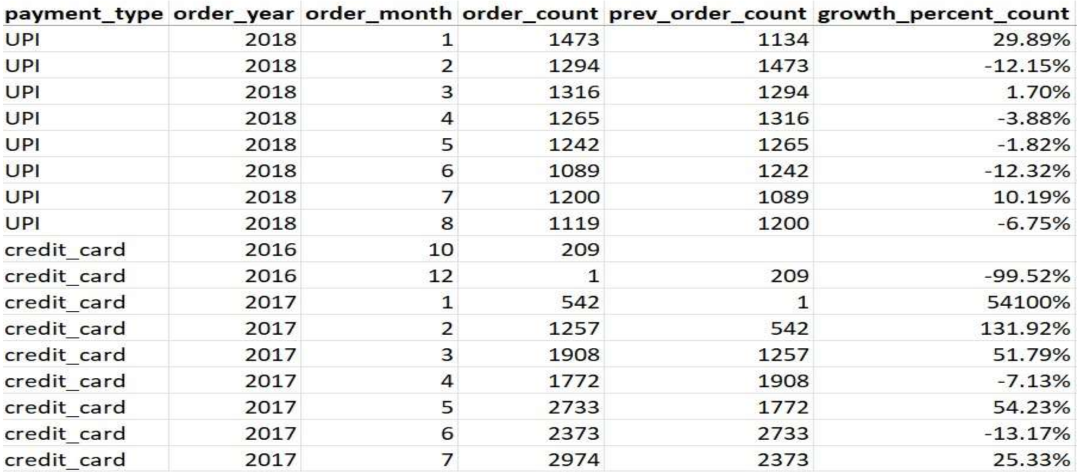
```
with table1 as
(
select
  payment_type,
  extract(year from order_purchase_timestamp) as order_year,
  extract(month from order_purchase_timestamp) as order_month,
  count(o.order_id) as order_count
from `sql_project.payments` p
join `sql_project.orders` o
on p.order_id = o.order_id
where o.order_status = "delivered"
group by payment_type,order_year,order_month
order by payment_type,order_year,order_month
),
table2 as
(
select
  *,
  lag(order_count) over(partition by payment_type order by payment_type,order_
year,order_month) as prev_order_count
from
  table1
)
select
  *,
  concat(round(((order_count-
prev_order_count)/prev_order_count)*100,2),"%") as growth_percent_count
from
  table2
order by payment_type,order_year,order_month
```

Query Screenshot:



```
with table1 as
(
select
  payment_type,
  extract(year from order_purchase_timestamp) as order_year,
  extract(month from order_purchase_timestamp) as order_month,
  count(o.order_id) as order_count
from `sql_project.payments` p
join `sql_project.orders` o
on p.order_id = o.order_id
where o.order_status = "delivered"
group by payment_type,order_year,order_month
order by payment_type,order_year,order_month
```

Result:



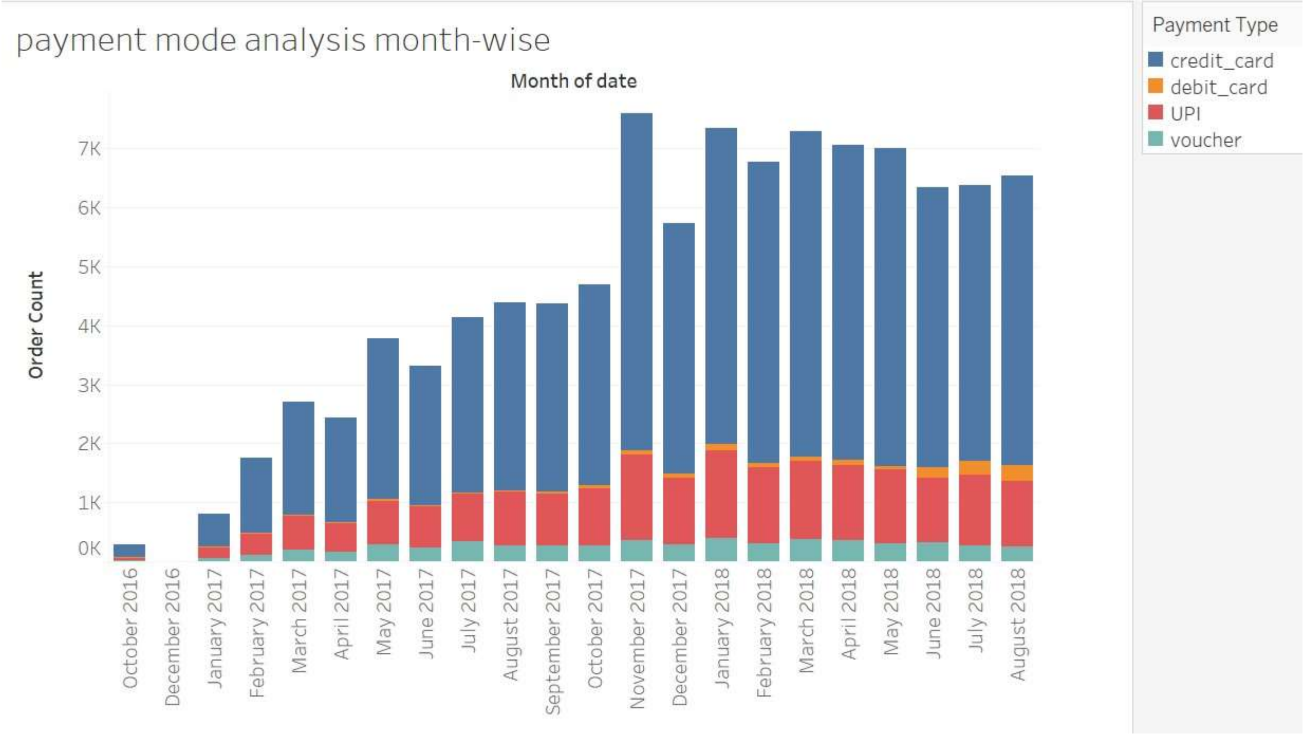
payment_type	order_year	order_month	order_count	prev_order_count	growth_percent_count
UPI	2018	1	1473	1134	29.89%
UPI	2018	2	1294	1473	-12.15%
UPI	2018	3	1316	1294	1.70%
UPI	2018	4	1265	1316	-3.88%
UPI	2018	5	1242	1265	-1.82%
UPI	2018	6	1089	1242	-12.32%
UPI	2018	7	1200	1089	10.19%
UPI	2018	8	1119	1200	-6.75%
credit_card	2016	10	209		
credit_card	2016	12	1	209	-99.52%
credit_card	2017	1	542	1	54100%
credit_card	2017	2	1257	542	131.92%
credit_card	2017	3	1908	1257	51.79%
credit_card	2017	4	1772	1908	-7.13%
credit_card	2017	5	2733	1772	54.23%
credit_card	2017	6	2373	2733	-13.17%
credit_card	2017	7	2974	2373	25.33%

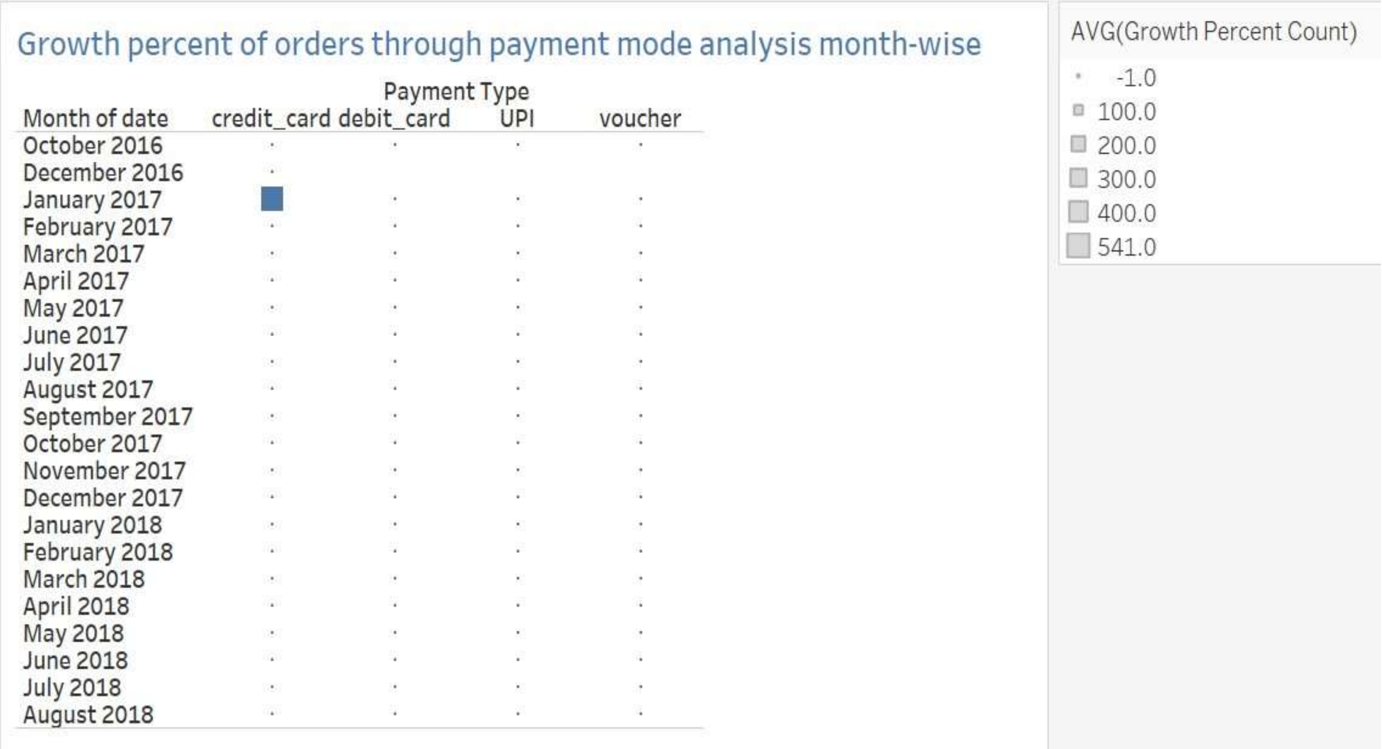
Insights:

Below two charts have been analysed, analysis of payment mode month-wise in all states and month-wise analysis of growth percent of orders through payment mode in all states.

It can be clearly seen that customers likely to use credit card as payment option followed by UPI payment. Very less customers use Voucher mode for payment.

A sharp peak in credit card payment mode can be seen in January 2017. In December 2016, only one customer has made the payment through credit card. However, in January 2017, 542 customers have paid for orders through credit card. This causes a sharp rise in growth percent usage of credit card.



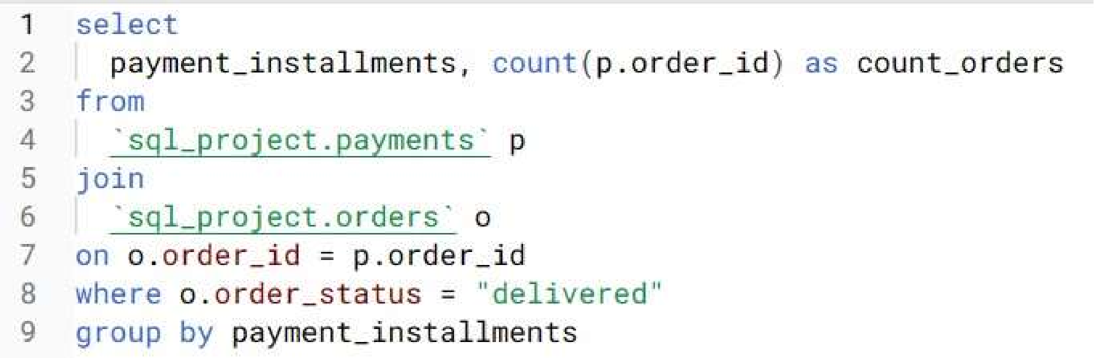


6.2. Count of orders based on the no. of payment instalments.

Query:

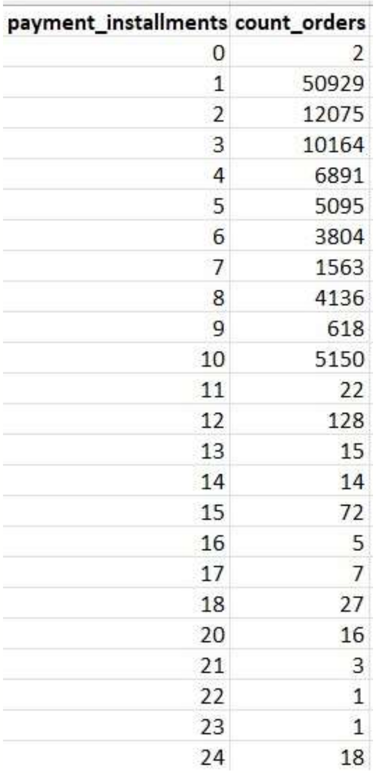
```
select
  payment_installments,
  count(p.order_id) as count_orders
from
  `sql_project.payments` p
join
  `sql_project.orders` o
on o.order_id = p.order_id
where o.order_status = "delivered"
group by payment_installments
```

Query Screenshot:



```
select
  payment_installments, count(p.order_id) as count_orders
from
  `sql_project.payments` p
join
  `sql_project.orders` o
on o.order_id = p.order_id
where o.order_status = "delivered"
group by payment_installments
```

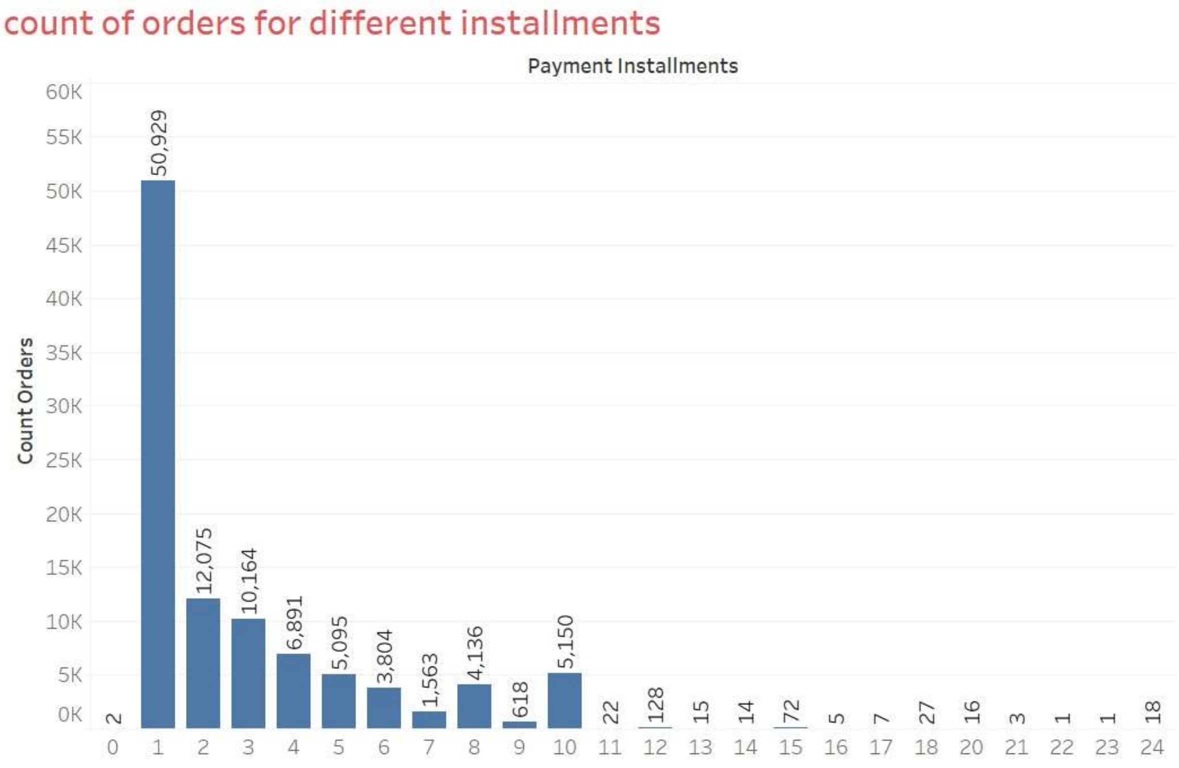
Result:



payment_installments	count_orders
0	2
1	50929
2	12075
3	10164
4	6891
5	5095
6	3804
7	1563
8	4136
9	618
10	5150
11	22
12	128
13	15
14	14
15	72
16	5
17	7
18	27
20	16
21	3
22	1
23	1
24	18

Insights:

Customers are likely to buy products on instalments preferably for one month-instalment. It can be observed that customers do not tend to hold the payment money for long months. Usually, they prefer to pay all amount within a year.



count of orders for different installments

ACTIONABLE INSIGHTS

Note: All insights have been shared with each problem statement respectively.

RECOMMENDATION

1. There are less number of customers in north region of Brazil. So, marketing needs to be done to increase the customer base.
2. As the number of orders are increasing month by month, company needs to make sure that there should be enough manpower and technical capabilities to avoid any kind of delay or customer grievances.
3. Customers are likely to order throughout a day. Hence, web services must be updated time to time to ensure customer attention.
4. In some states like Sao Paulo, Rio De Janeiro and Minas Gerais, counts are increasing at good rate in comparison with other states.
   Company must take feedback ratings or reviews from other states and based on that, Company should launch some schemes or improve customer services to attract customers from other states as well.
5. It has been observed that percent increase in cost incurred on orders is quite greater than the percent increase in order payments by customers. To stabilize this condition, company must take a deep insight on cost incurred on price of orders, freight value spend on that order and other factors as well.
6. It has been analysed that in some states like *AL,MA,SE,ES* and *BA,* the time taken for order delivery is too high. Several factors need to be taken into account for analysing the reason for slow delivery:
   - Is the state region rural?
   - The lifestyle of customers in that region.
   - Population in that region.
   - Freight values in that region.
7. It can be clearly seen that customers likely to use credit card as payment option followed by UPI payment. Therefore, company needs to have multiple payment gateways and supports payment modes of all existing banks.
8. Customers are likely to buy products on instalments preferably for one month instalment. To lure more customers for order, instalment scheme with 0 percent interest can be introduced.